

EDA技术及其应用

第3章 宏功能模块应用

3.1 流水线乘法累加器设计

3.1.1 电路结构与工作原理

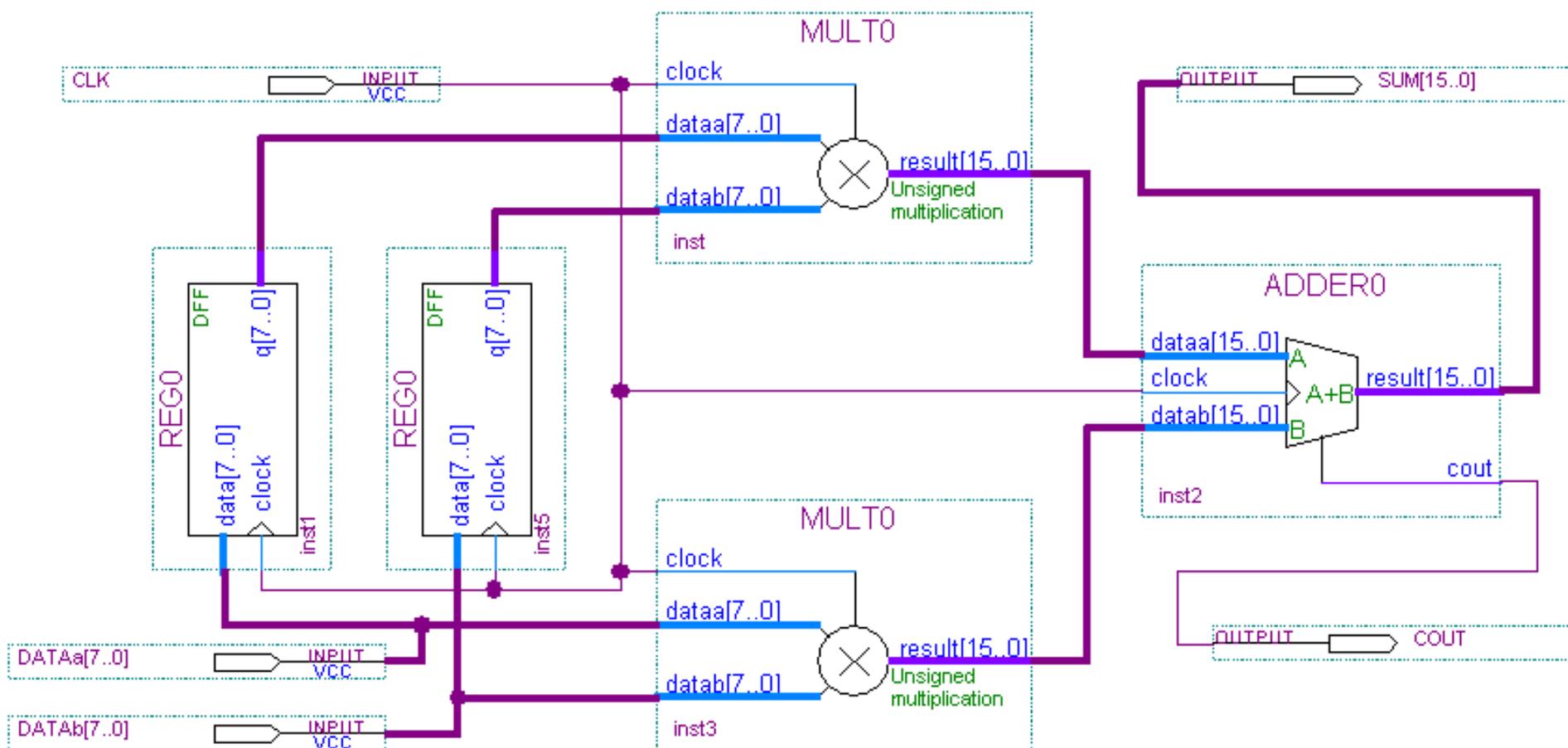


图3-1 流水线乘法累加器顶层设计

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

1. 调用乘法器

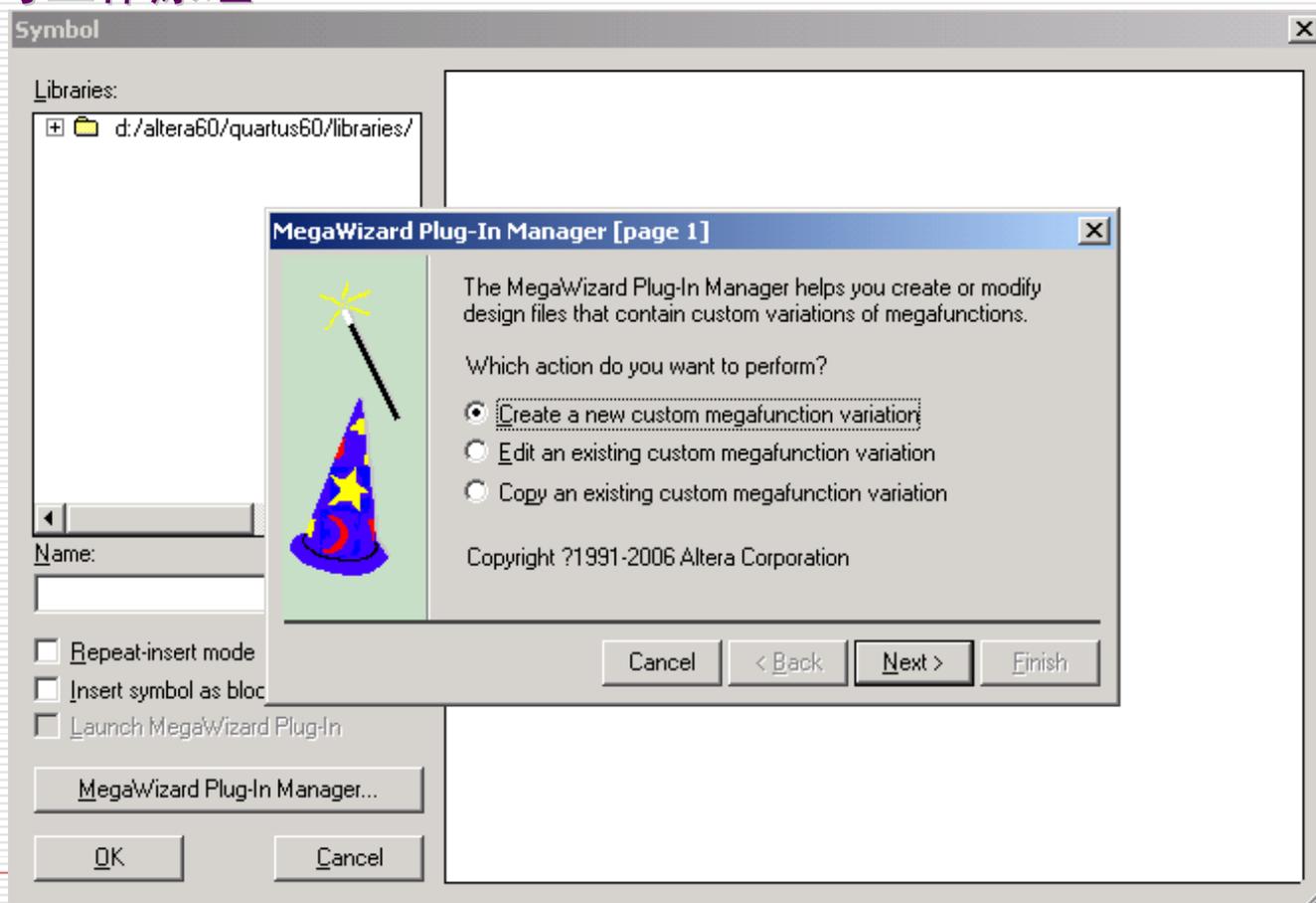
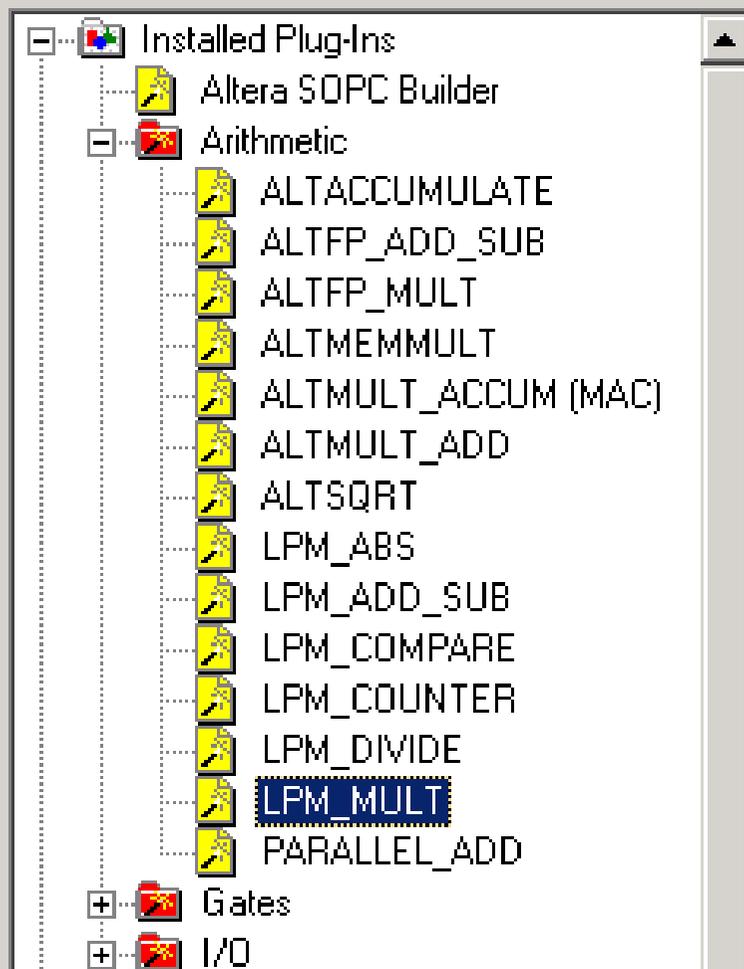


图3-2 定制新的宏功能块

Which megafunction would you like to customize?
Select a megafunction from the list below



A tree view showing installed plug-ins. The 'Arithmetic' folder is expanded, listing various megafunctions. 'LPM_MULT' is selected and highlighted with a blue background.

- Installed Plug-Ins
 - Altera SOPC Builder
 - Arithmetic
 - ALTACCUMULATE
 - ALTFP_ADD_SUB
 - ALTFP_MULT
 - ALTMEMMULT
 - ALTMULT_ACCUM (MAC)
 - ALTMULT_ADD
 - ALTSQRT
 - LPM_ABS
 - LPM_ADD_SUB
 - LPM_COMPARE
 - LPM_COUNTER
 - LPM_DIVIDE
 - LPM_MULT**
 - PARALLEL_ADD
 - Gates
 - I/O

Which device family will you be using?

Cyclone II

Which type of output file do you want to create?

- AHDL
- VHDL
- Verilog HDL

What name do you want for the output file?

Browse...

D:\MULADD\MULT0

- Generate clear box netlist file instead of a default wrapper file (for use with supported EDA synthesis tools only)
- Return to this page for another create operation

Note: To compile a project successfully in the Quartus II software, your design files must be in the project directory, in the global user libraries specified in the Options dialog box (Tools menu), or a user library specified in the User Libraries page of the Settings dialog box (Assignments menu).

Your current user library directories are:

图3-3 选择LPM宏功能模块



LPM_MULT

Version 6.0

[About](#)[Documentation](#)

1 Parameter Settings

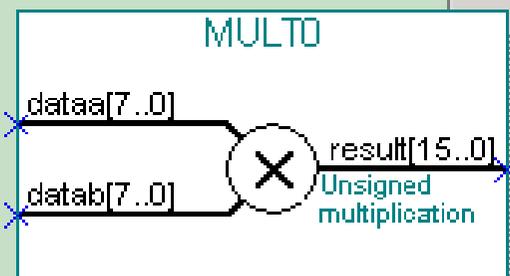
2 Simulation Library

3 Summary Page

General

General 2

Pipelining



Resource Usage

117 lut

Multiplier configuration

- Multiply 'dataa' input by 'datab' input
- Multiply 'dataa' input by itself (squaring operation)

How wide should the 'dataa' input bus be?

8 bits

How wide should the 'datab' input bus be?

8 bits

 Create a 'sum' input bus with a width of

1 bits

How should the width of the 'result' output be determined?

- Automatically calculate the width
- Restrict the width to 16 bits

Cancel

< Back

Next >

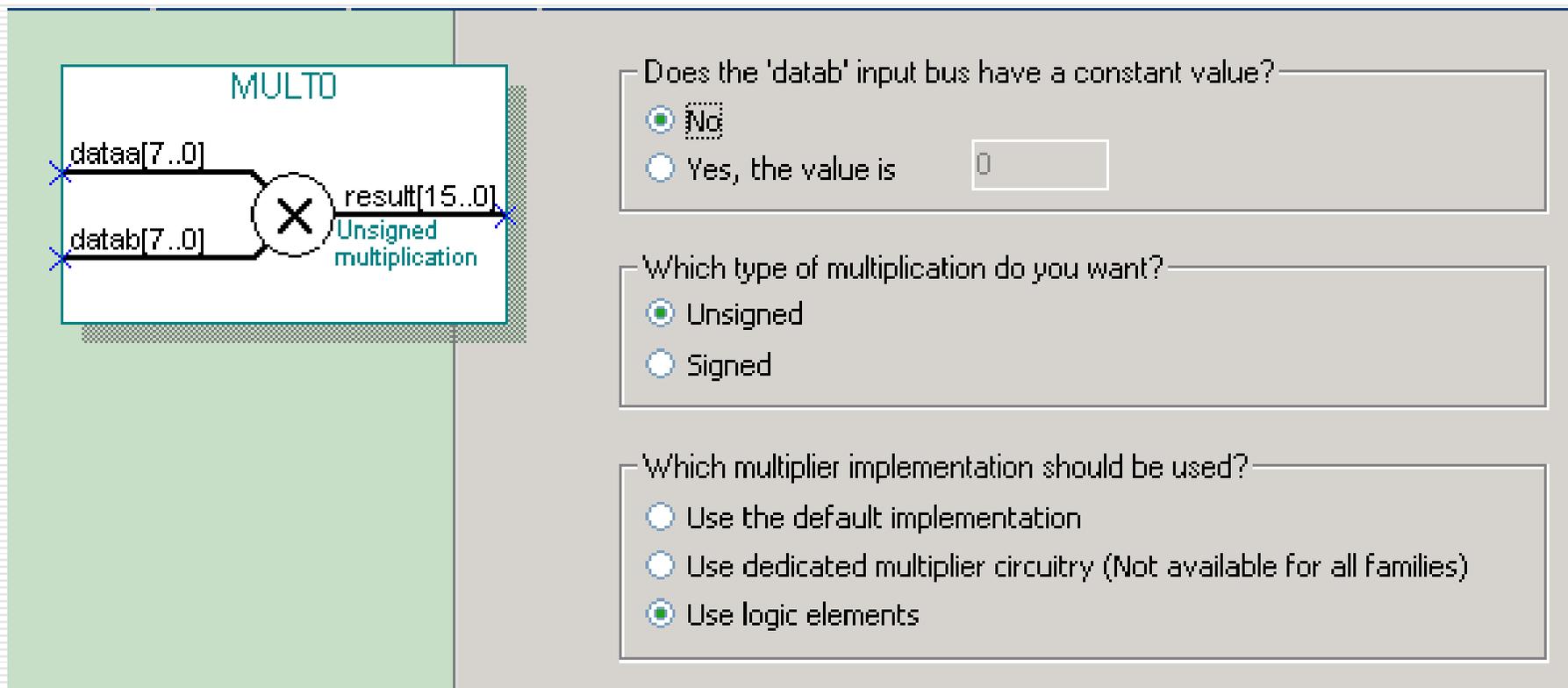
Finish

图3-4 设置乘法器参数

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

1. 调用乘法器



The image shows a configuration window for a multiplier component. On the left, a circuit diagram labeled 'MULTO' shows two 8-bit inputs, 'dataa[7..0]' and 'datab[7..0]', connected to a multiplier symbol (a circle with an 'X'). The output is 'result[15..0]'. Below the multiplier symbol, the text 'Unsigned multiplication' is displayed. On the right, there are three configuration panels:

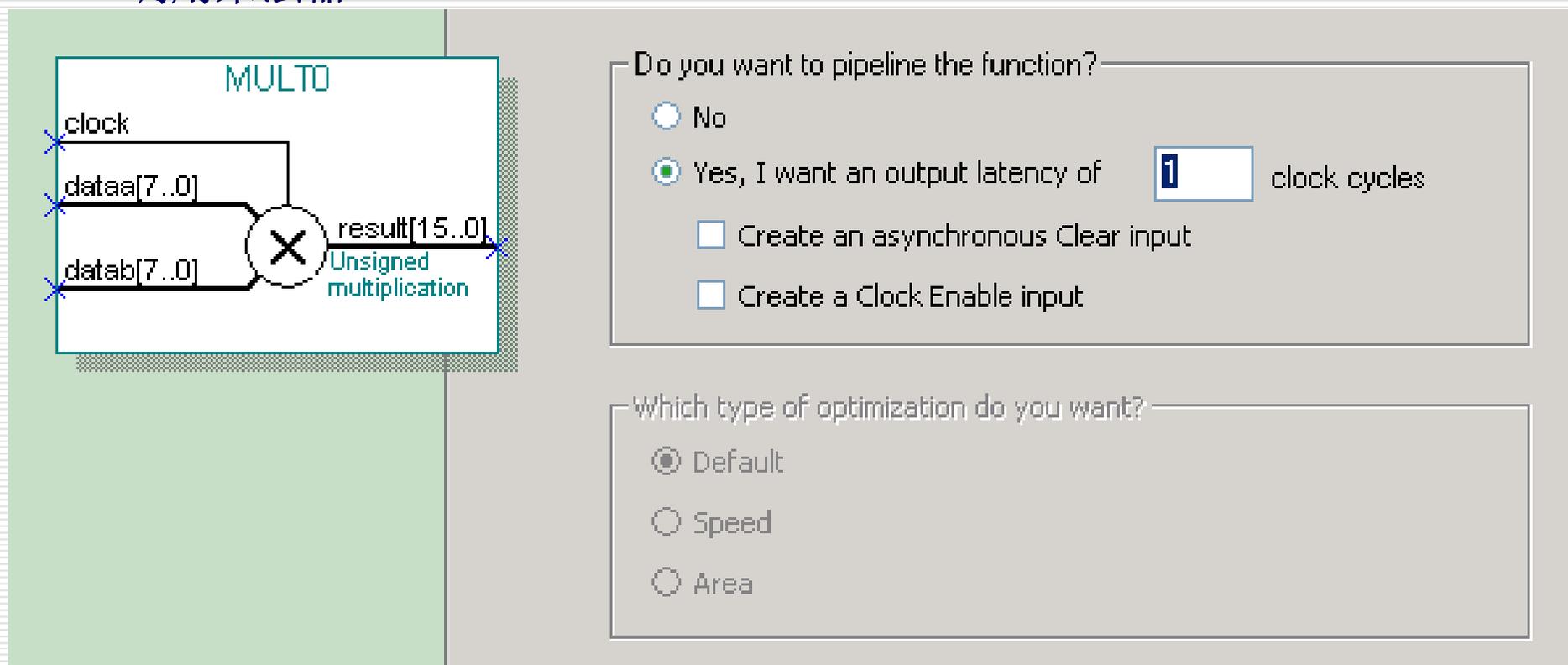
- Does the 'datab' input bus have a constant value?**
 - No
 - Yes, the value is
- Which type of multiplication do you want?**
 - Unsigned
 - Signed
- Which multiplier implementation should be used?**
 - Use the default implementation
 - Use dedicated multiplier circuitry (Not available for all families)
 - Use logic elements

图3-5 设置乘法器结构类型

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

1. 调用乘法器



The image shows a configuration window for the LPM_MULT0 multiplier component. On the left, a schematic diagram shows the multiplier with inputs 'clock', 'dataa[7..0]', and 'datab[7..0]', and an output 'result[15..0]'. The multiplier is labeled 'Unsigned multiplication'. On the right, the configuration options are as follows:

Do you want to pipeline the function?

- No
- Yes, I want an output latency of clock cycles
- Create an asynchronous Clear input
- Create a Clock Enable input

Which type of optimization do you want?

- Default
- Speed
- Area

图3-6将LPM乘法器设置为流水线工作方式

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

2. 调用加法器和锁存器

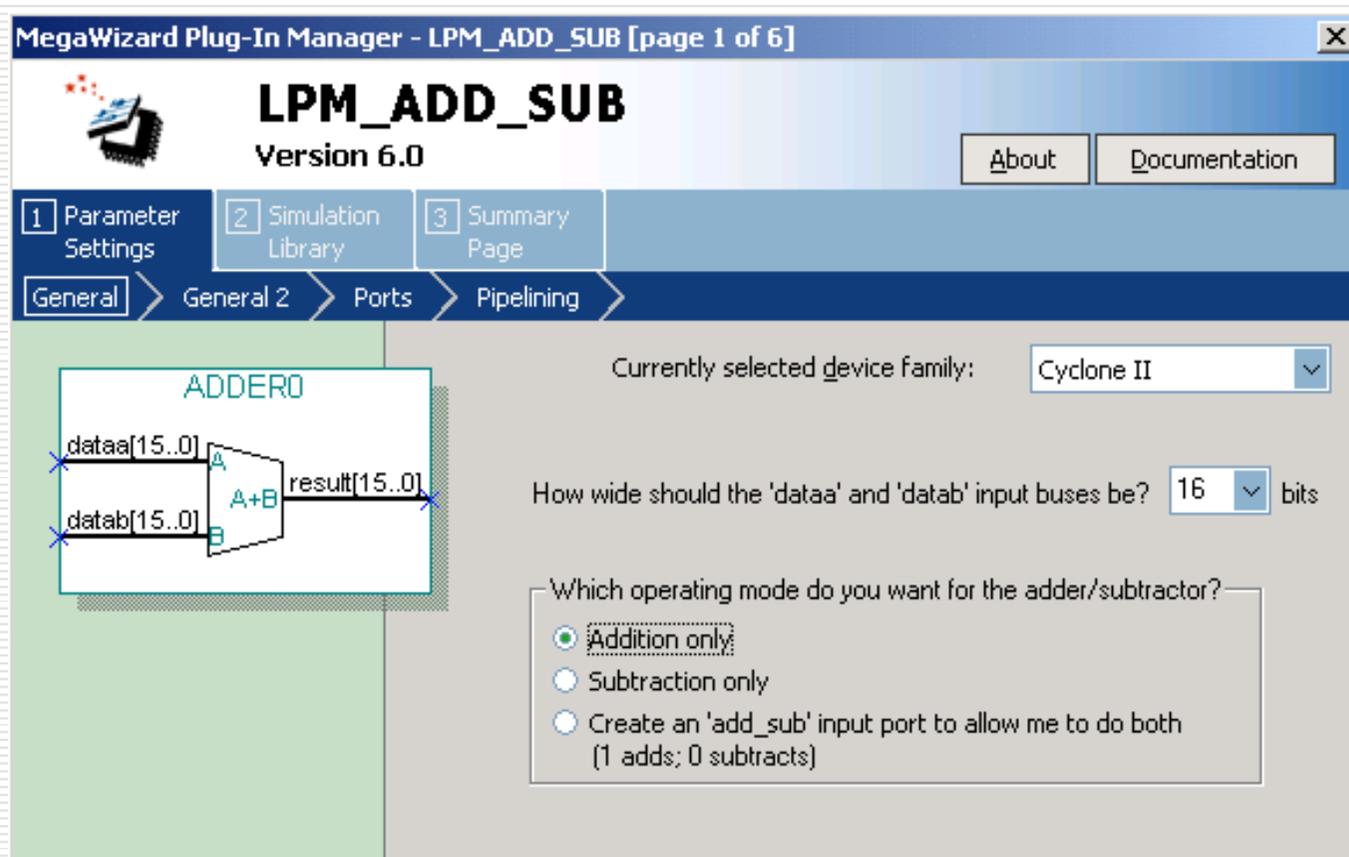


图3-7 设置LPM加法器类型

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

2. 调用加法器和锁存器

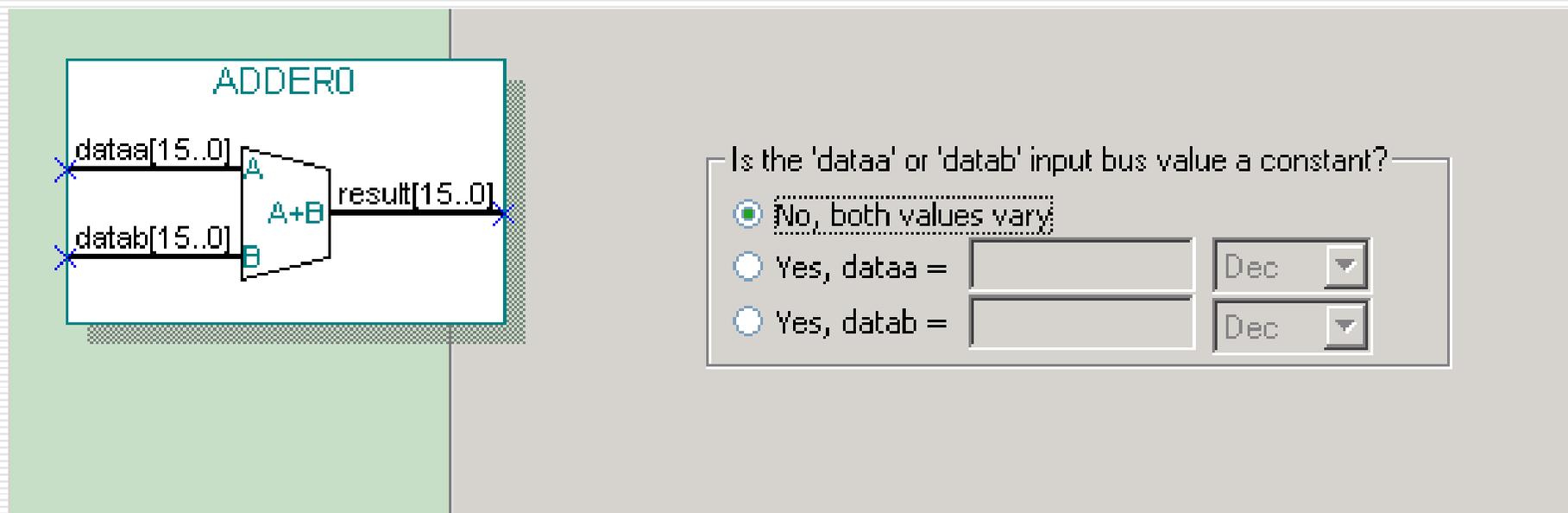


图3-8 选择加法器数据输入类型

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

2. 调用加法器和锁存器

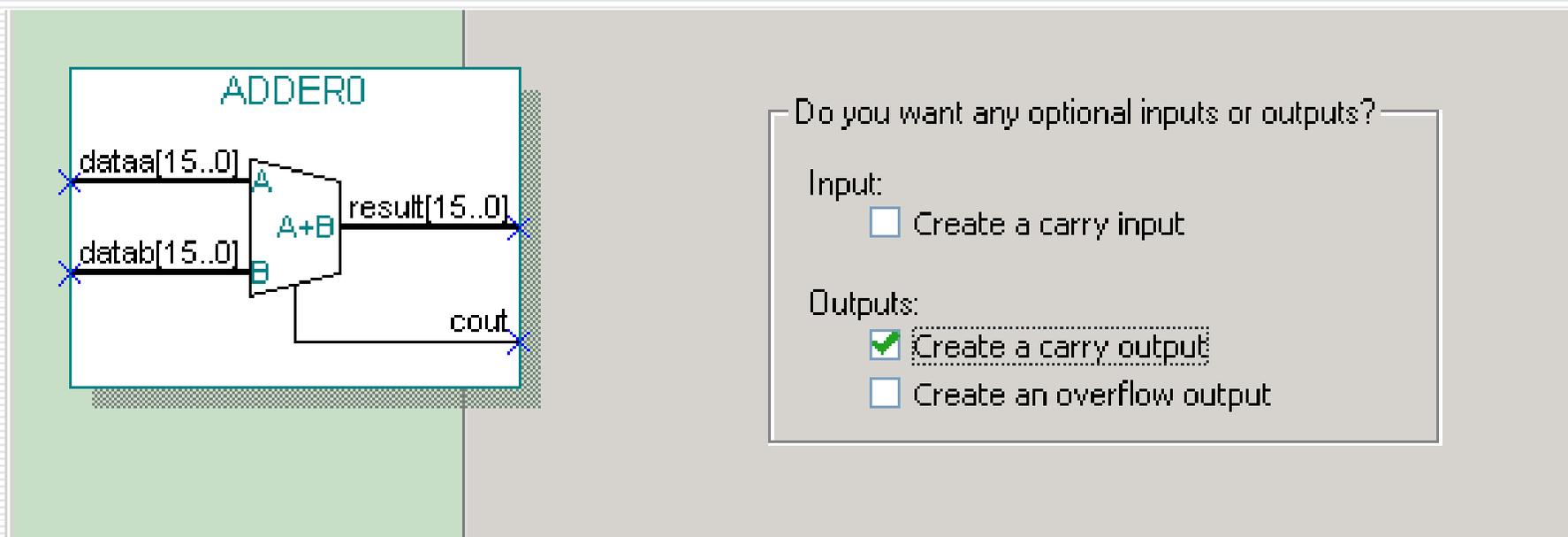


图3-9 为加法器增加进位输出

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

2. 调用加法器和锁存器

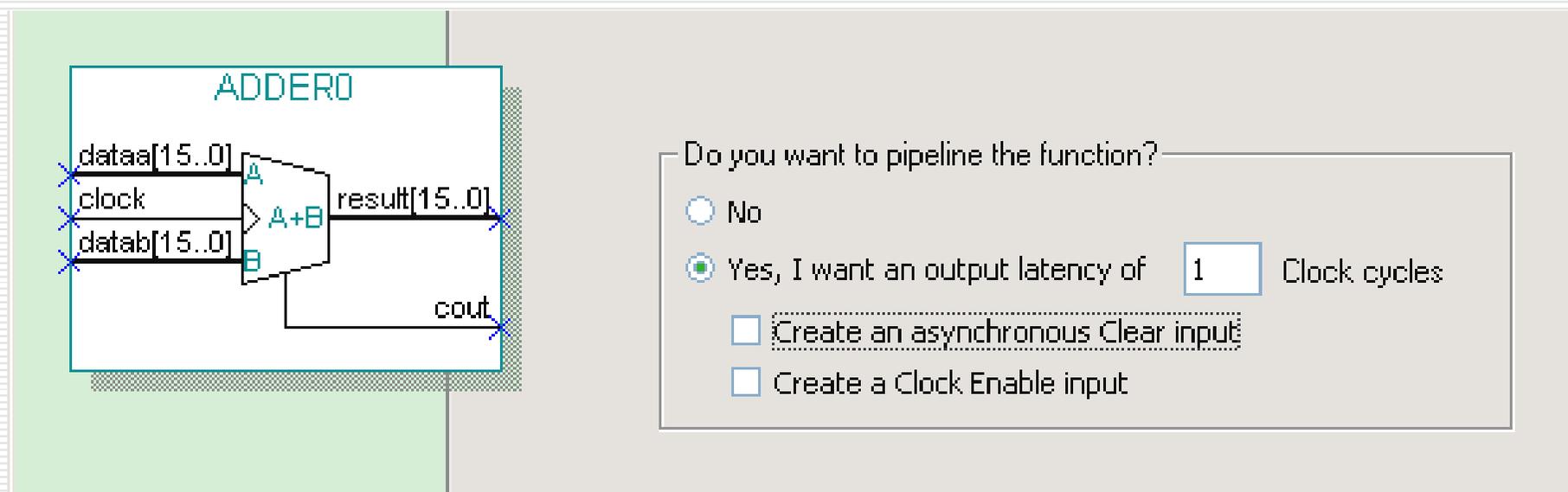


图3-10 为加法器增加流水线功能

3.1 流水线乘法累加器设计

3.1.2 电路结构与工作原理

2. 调用加法器和锁存器

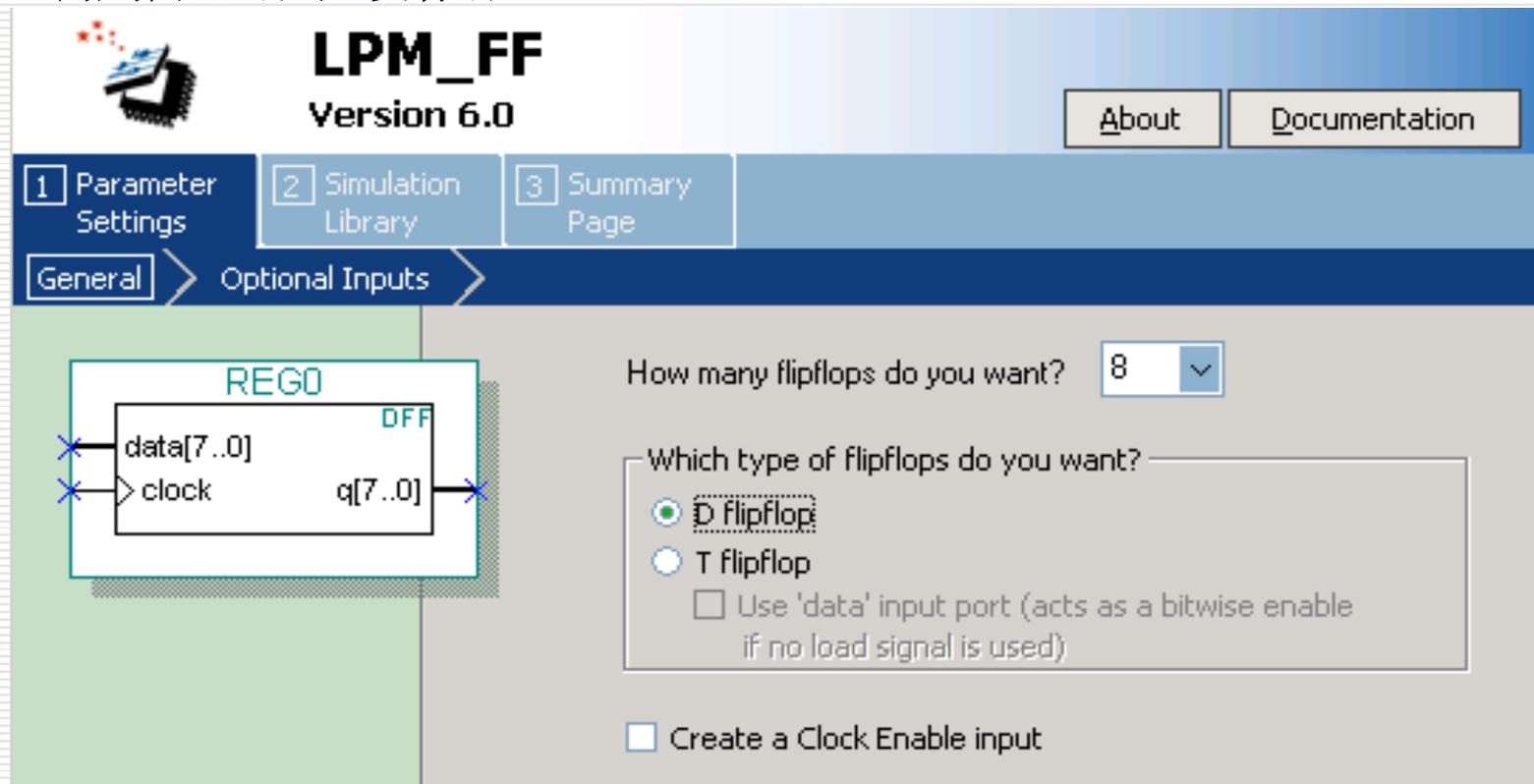


图3-11 为LPM寄存器选择D触发器类型

3.1 流水线乘法累加器设计

3.1.3 电路时序仿真与测试

```
Flow Status                Successful - Thu Aug
Quartus II Version         6.0 Build 202 06/20/
Revision Name              MULTADD
Top-level Entity Name      MULTADD
Family                     Cyclone II
Device                     EP2C8Q208C8
Timing Models              Final
Met timing requirements     Yes
Total logic elements       223 / 8,256 ( 3 % )
Total registers            127
Total pins                 34 / 138 ( 25 % )
Total virtual pins         0
Total memory bits          0 / 165,888 ( 0 % )
Embedded Multiplier 9-bit elements 0 / 36 ( 0 % )
Total PLLs                 0 / 2 ( 0 % )
```

图3-12 基于逻辑宏单元的设计报告

3.1 流水线乘法累加器设计

3.1.3 电路时序仿真与测试

Flow Status	Successful - Thu Aug
Quartus II Version	6.0 Build 202 06/20/2
Revision Name	MULTADD
Top-level Entity Name	MULTADD
Family	Cyclone II
Device	EP2C8Q208C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	18 / 8,256 (< 1 %)
Total registers	17
Total pins	34 / 138 (25 %)
Total virtual pins	0
Total memory bits	0 / 165,888 (0 %)
Embedded Multiplier 9-bit elements	2 / 36 (6 %)
Total PLLs	0 / 2 (0 %)

图3-13 基于专用嵌入式乘法器模块的设计报告

3.1 流水线乘法累加器设计

3.1.3 电路时序仿真与测试

Figure 3-14 shows a screenshot of a software interface displaying a Timing Analyzer Summary report for a project named MULTADD.bdf. The report is titled 'Compilation Report - Timing Analyzer Sum...' and is presented in a table format. The table has five columns: Type, Slack, Required Time, Actual Time, and From. The data rows are as follows:

Type	Slack	Required Time	Actual Time	From
1 Worst-case tsu	N/A	None	10.350 ns	DATAB[2]
2 Worst-case tco	N/A	None	9.806 ns	ADDER0:in
3 Worst-case th	N/A	None	-1.231 ns	DATAa[14]
4 Clock Setup: 'CLK'	N/A	None	140.90 MHz (period = 7.097 ns)	MULT0:inst
5 Total number of failed paths				

图3-14 基于逻辑宏单元的流水线乘法累加器时序分析报告

3.1 流水线乘法累加器设计

3.1.3 电路时序仿真与测试

The screenshot displays the 'Timing Analyzer Summary' window. The left pane shows a project tree with folders for 'Compilation Report', 'Analysis & Synthesis', 'Fitter', 'Assembler', and 'Timing Analyzer'. The 'Timing Analyzer' folder is expanded, showing 'Summary', 'Settings', and 'Clock Settings Summary'. The main pane shows a table with the following data:

Type	Slack	Required Time	Actual Time	From
1 Worst-case tsu	N/A	None	8.809 ns	DATAa[
2 Worst-case tco	N/A	None	8.981 ns	ADDER
3 Worst-case th	N/A	None	-0.829 ns	DATAb[
4 Clock Setup: 'CLK'	N/A	None	Restricted to 180.57 MHz (period = 5.538 ns)	REG0:ir
5 Total number of failed paths				

图3-15 基于专用嵌入式乘法器模块的流水线乘法累加器时序分析报告

3.1 流水线乘法累加器设计

3.1.3 电路时序仿真与测试

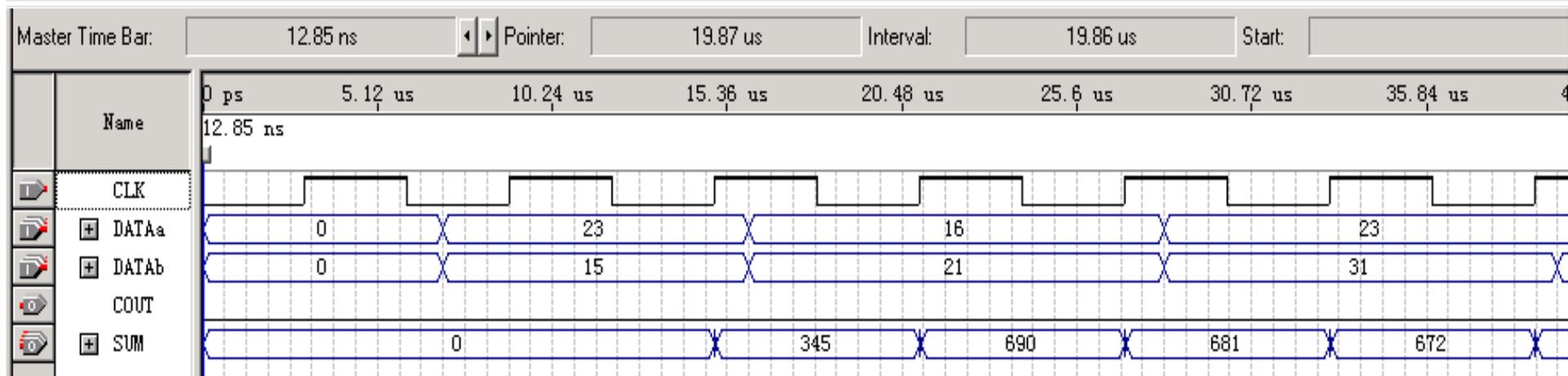


图3-16 MULTADD工程仿真波形

3.2 逻辑数据采样电路设计

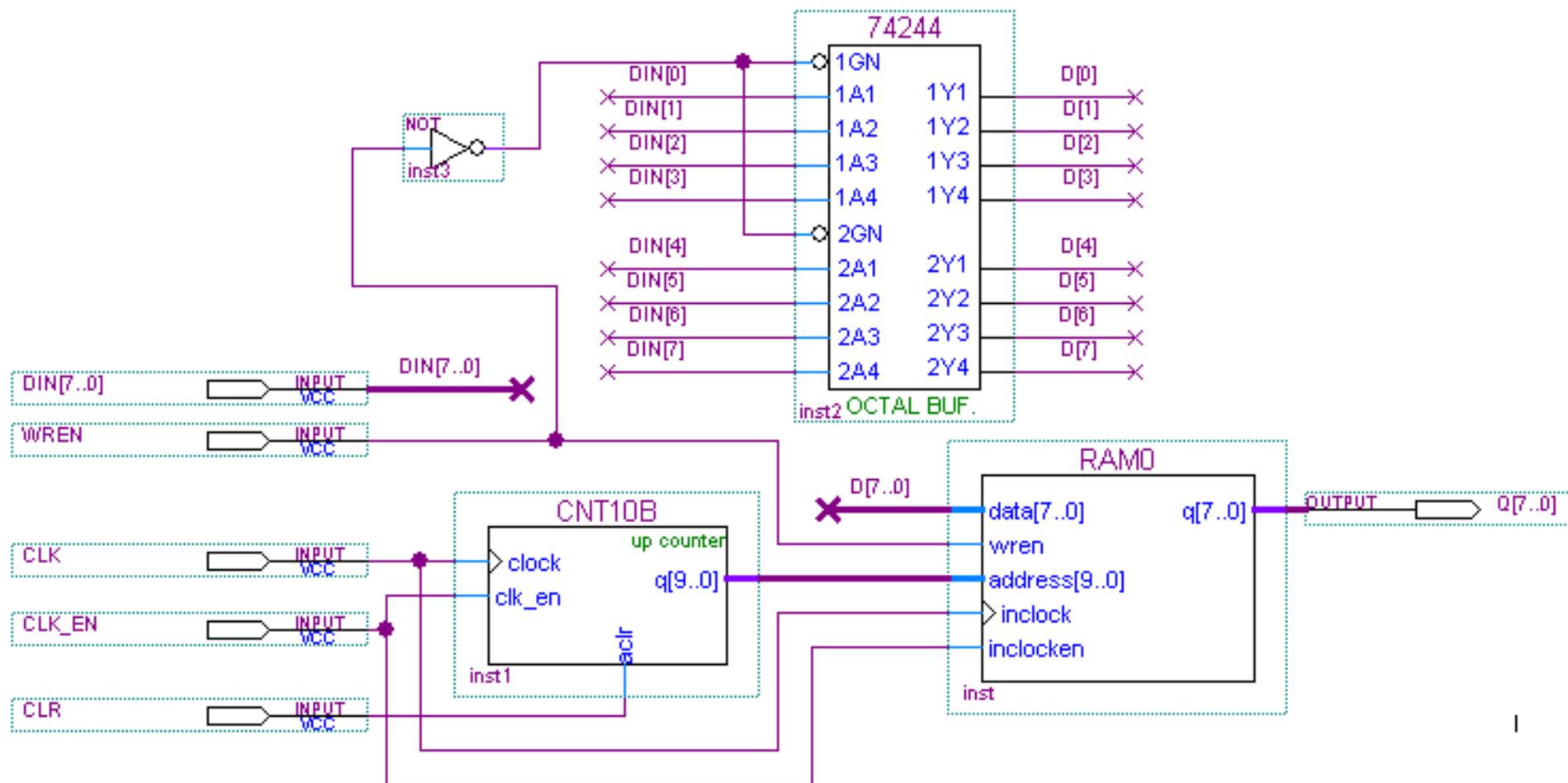


图3-17 逻辑数据采样电路顶层设计

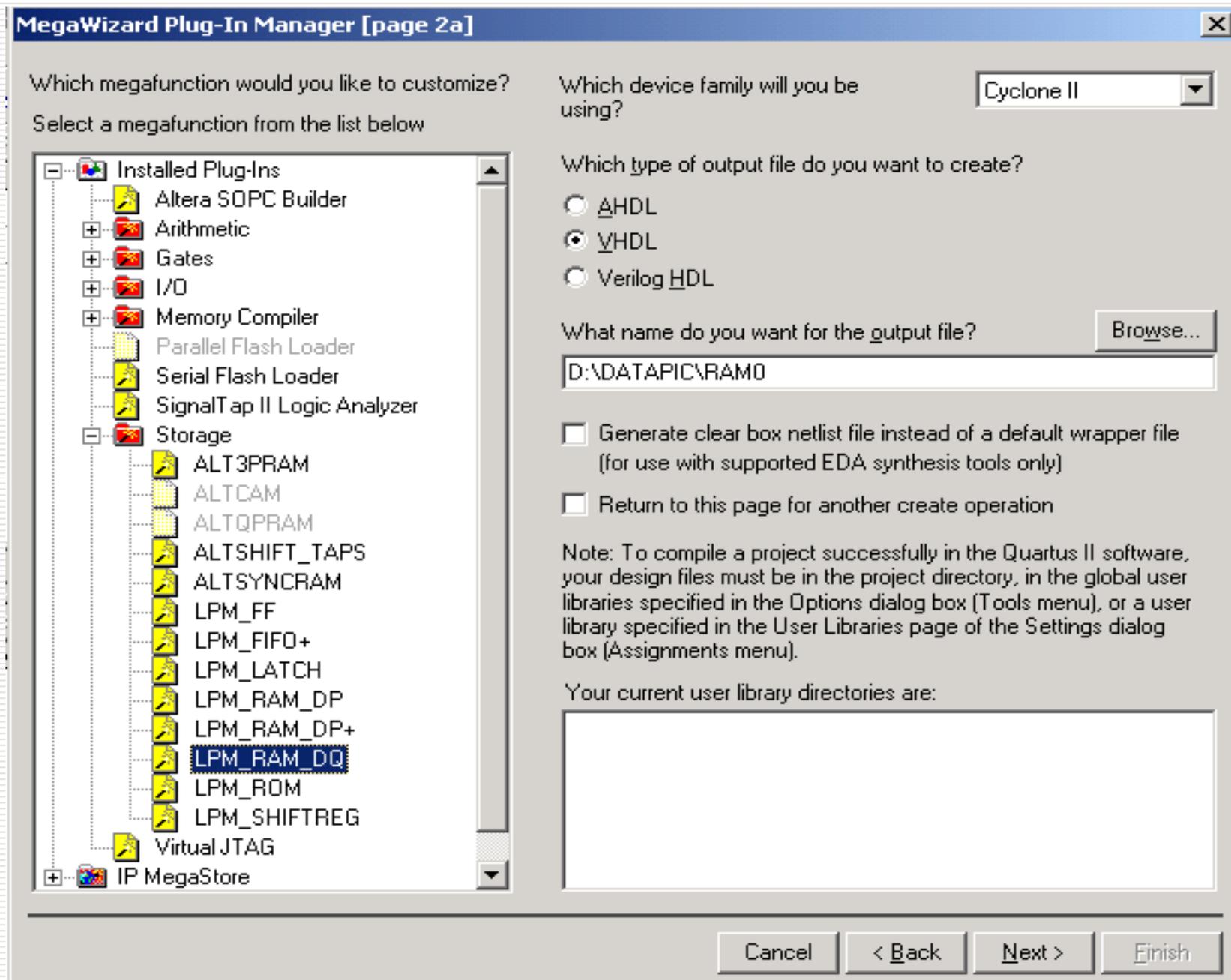


图3-18 调用LPM RAM宏功能模块

3.2 逻辑数据采样电路设计

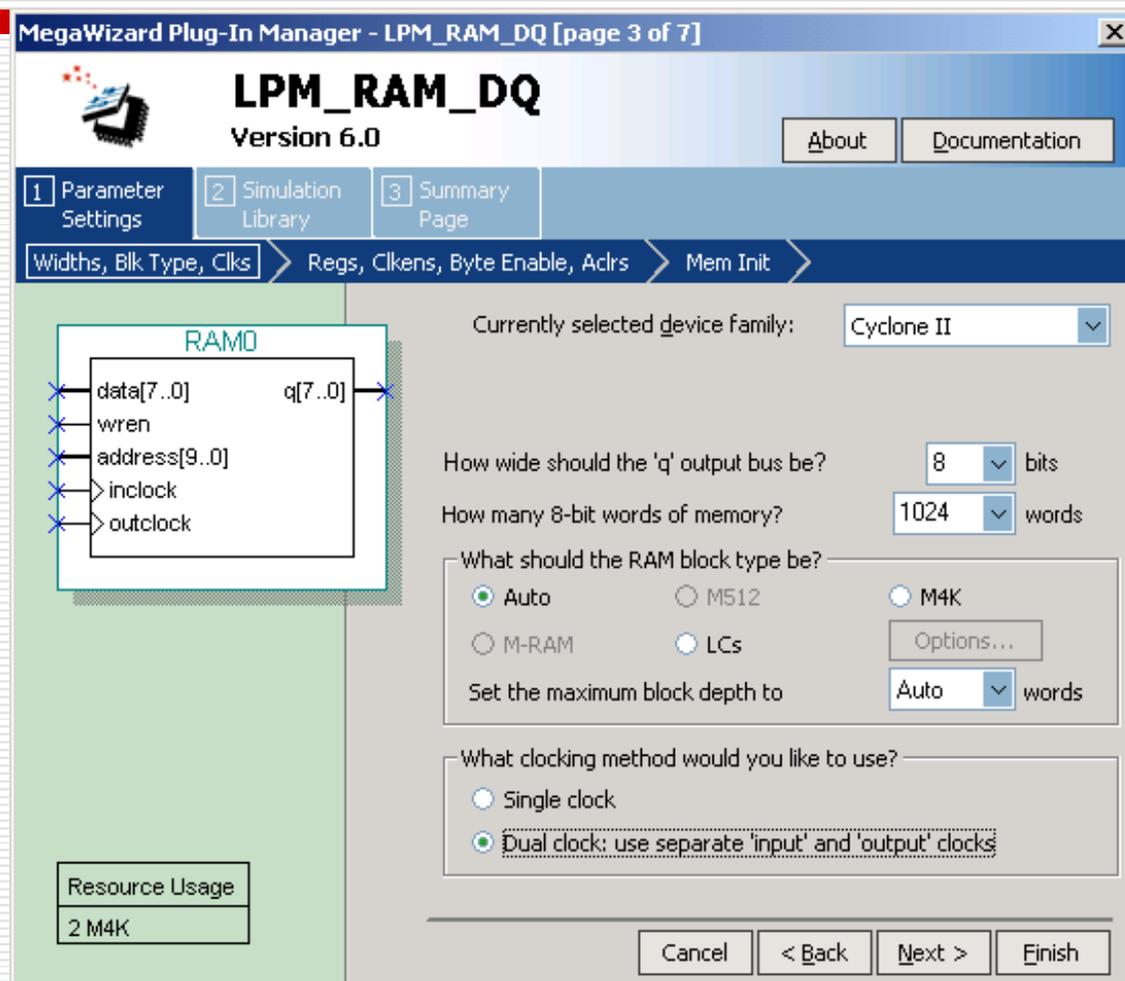
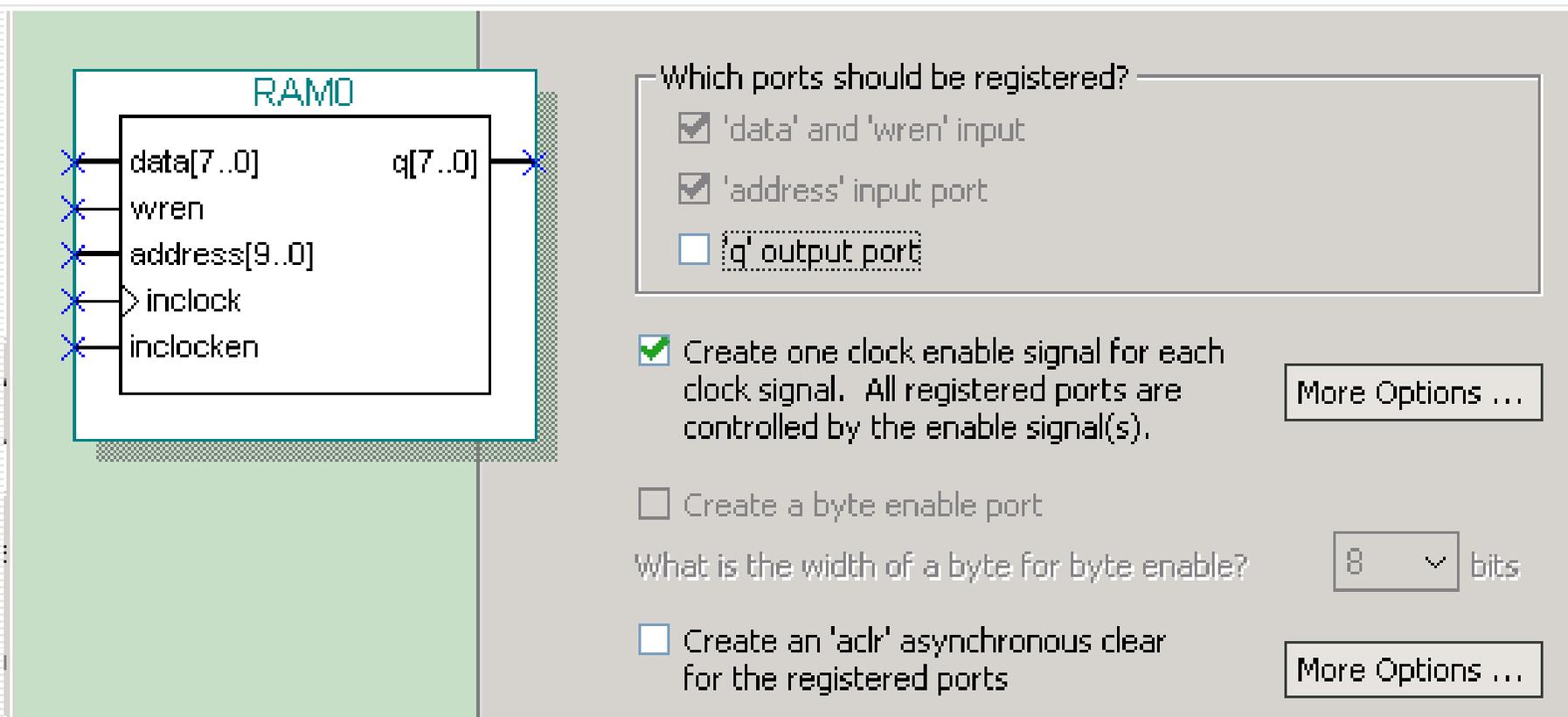


图3-19 LPM RAM参数设置

3.2 逻辑数据采样电路设计



The image shows a configuration window for a RAM component named "RAM0". On the left, the component's ports are listed: data[7..0] (output), wren (write enable), address[9..0] (input), inclock (clock), and inclocken (clock enable). On the right, there are several configuration options:

- Which ports should be registered?**
 - 'data' and 'wren' input
 - 'address' input port
 - 'q' output port
- Create one clock enable signal for each clock signal. All registered ports are controlled by the enable signal(s). [More Options ...](#)
- Create a byte enable port
- What is the width of a byte for byte enable? bits
- Create an 'aclr' asynchronous clear for the registered ports [More Options ...](#)

图3-20 增加时钟使能控制

3.2 逻辑数据采样电路设计

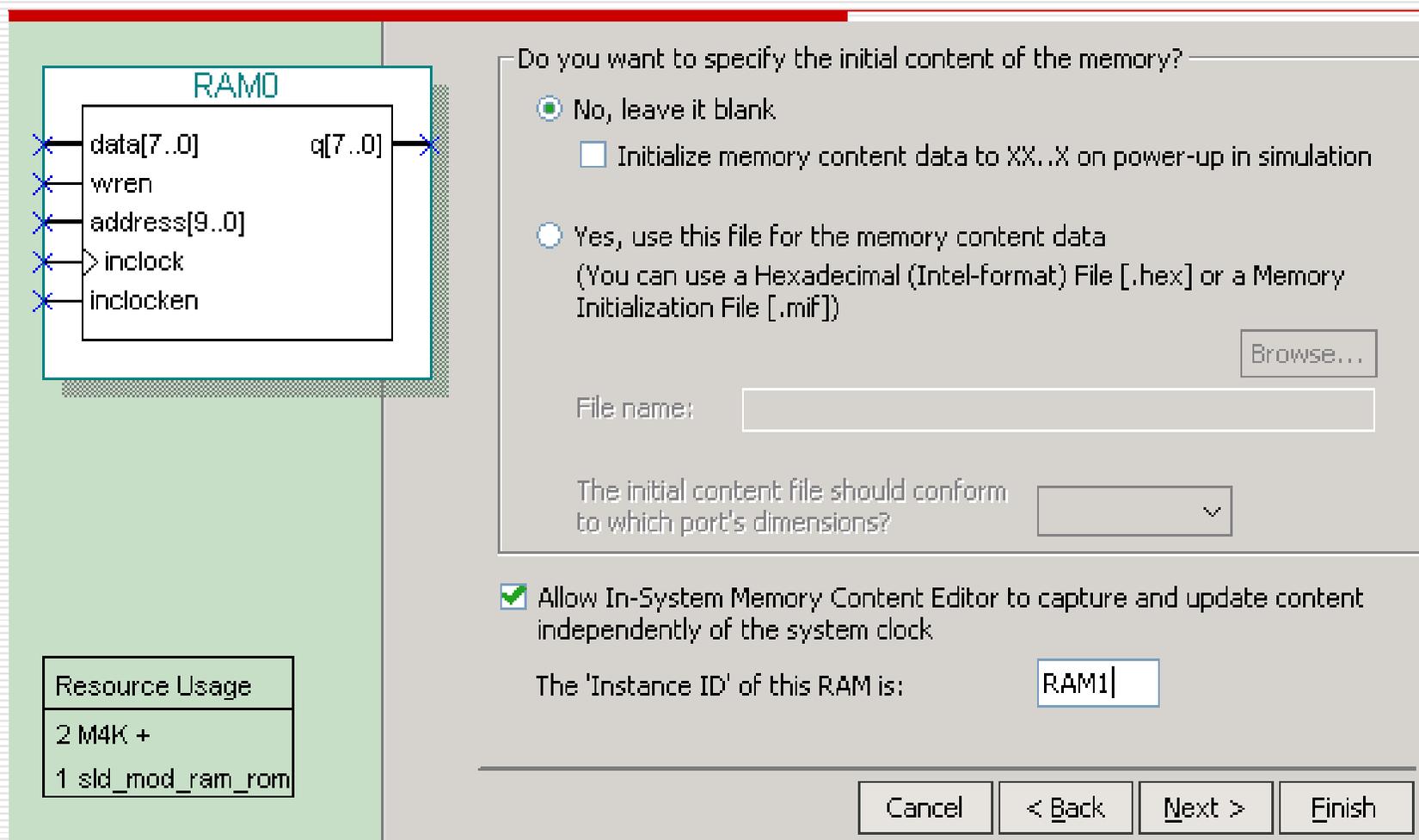


图3-21 允许在系统存储器内容编辑器能对此RAM编辑

3.2 逻辑数据采样电路设计

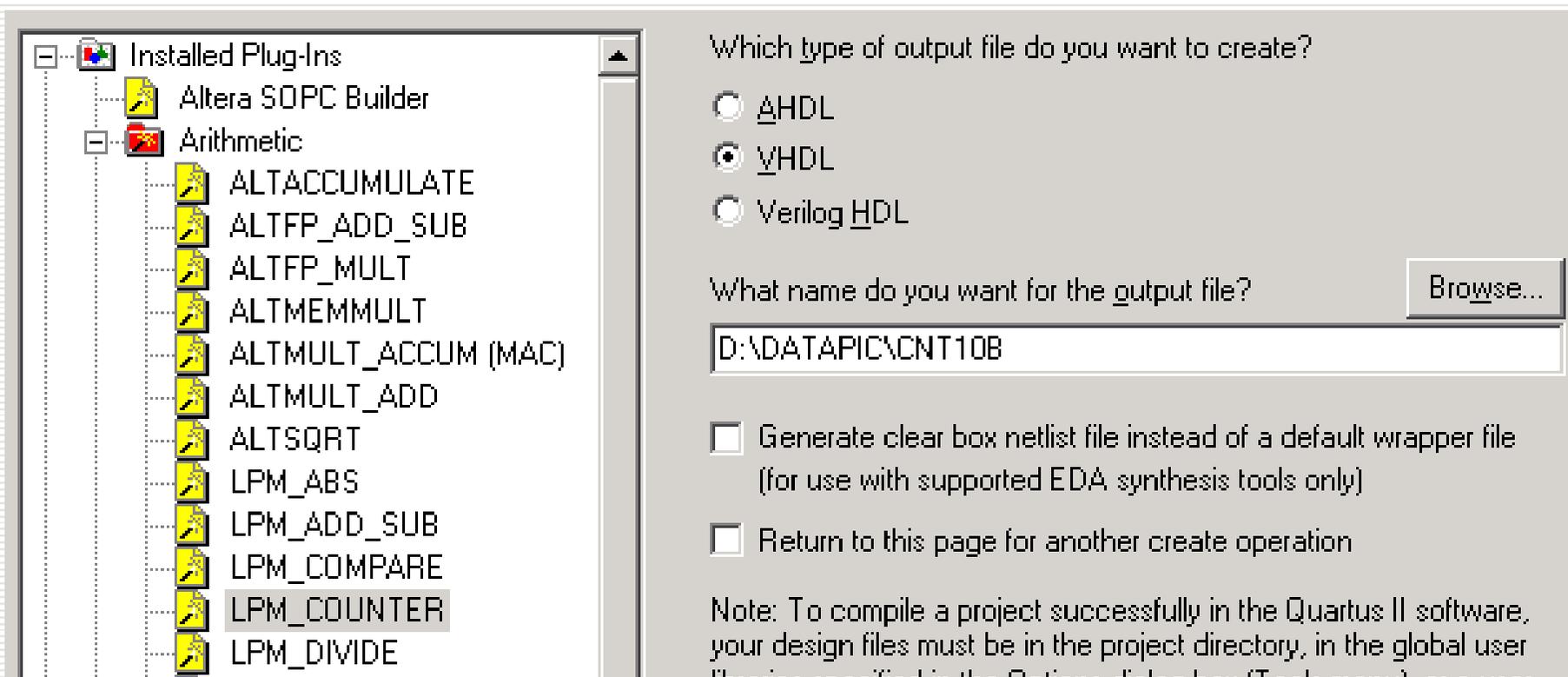


图3-22 调用LPM计数器

3.2 逻辑数据采样电路设计

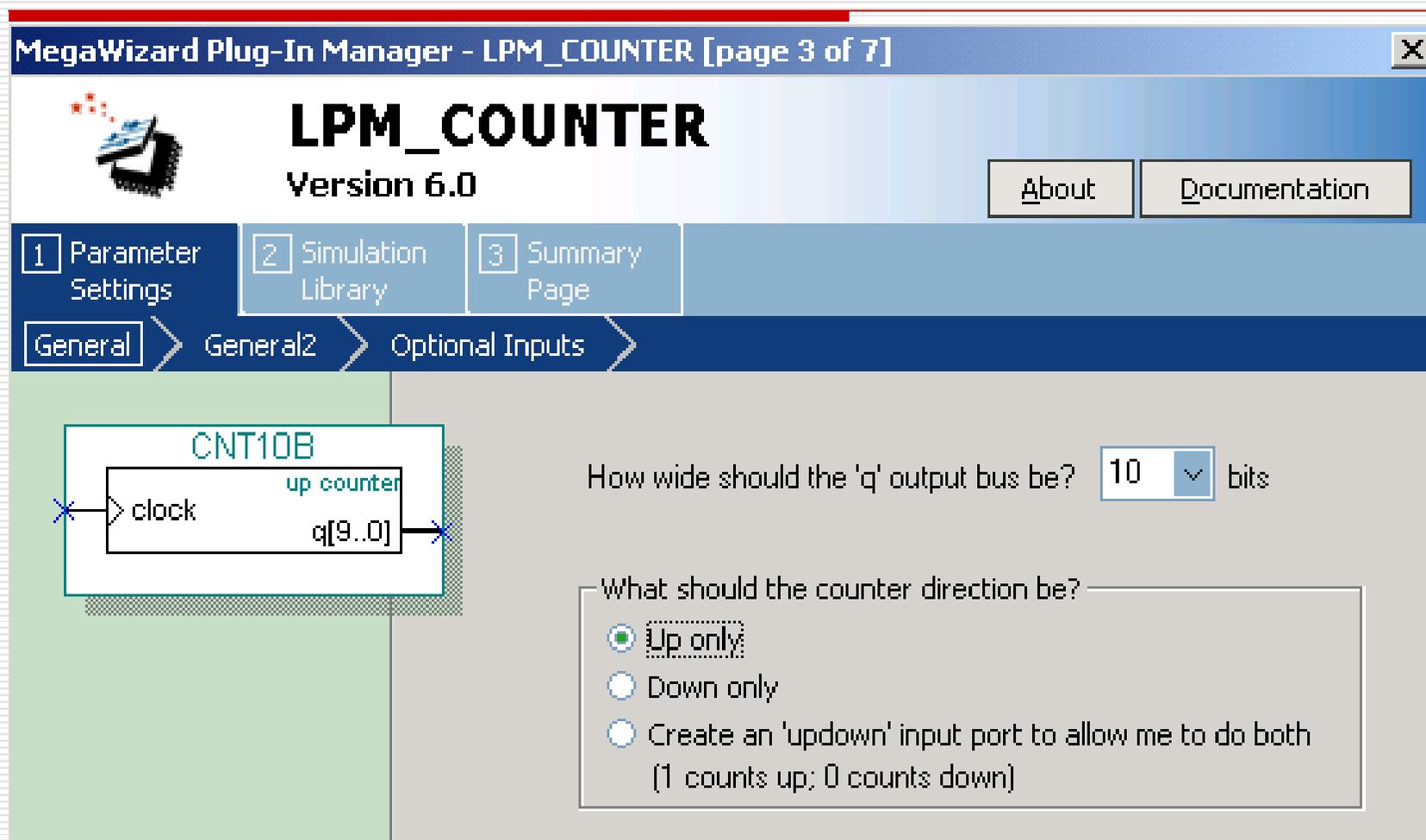


图3-23 设置为加法计数器

3.2 逻辑数据采样电路设计

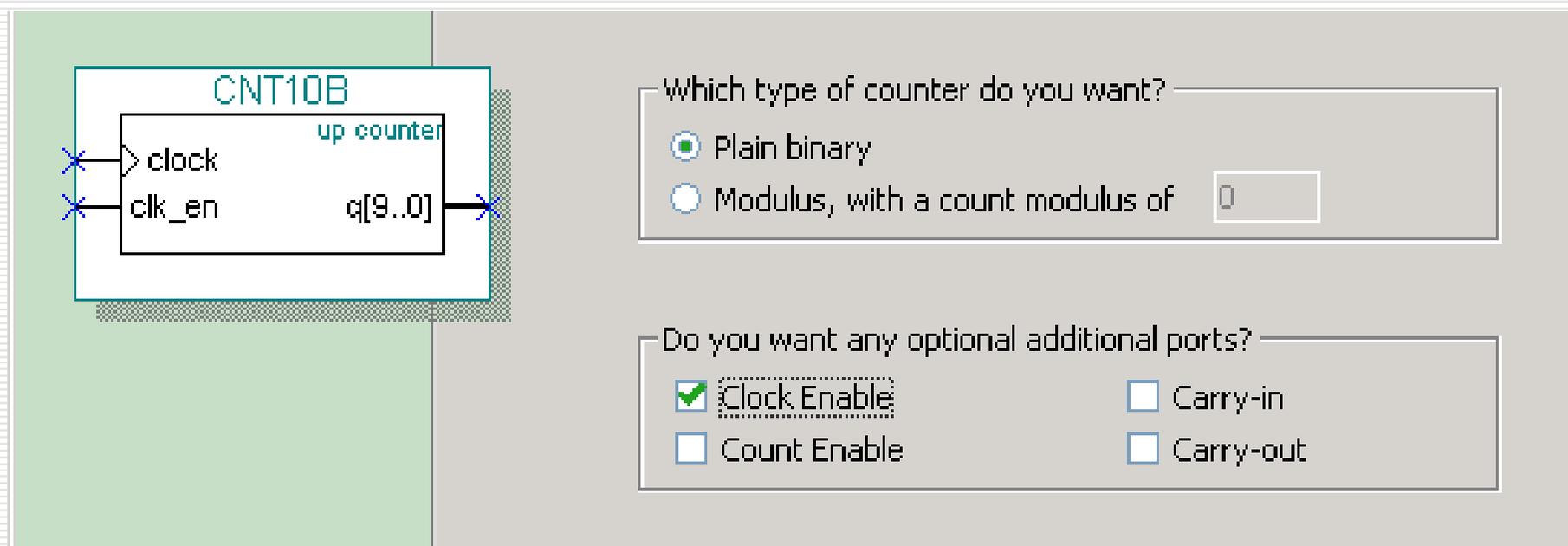


图3-24 设置为二进制计数器

3.2 逻辑数据采样电路设计

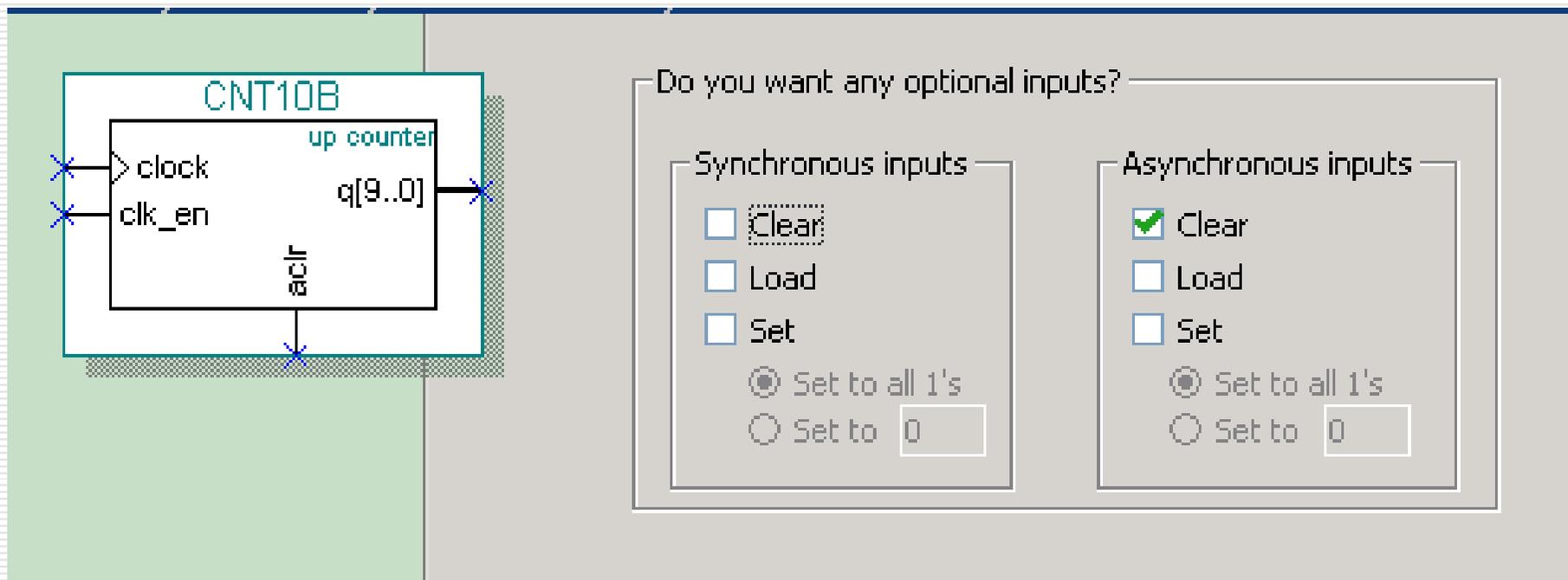


图3-25 增加异步清0控制

3.2 逻辑数据采样电路设计

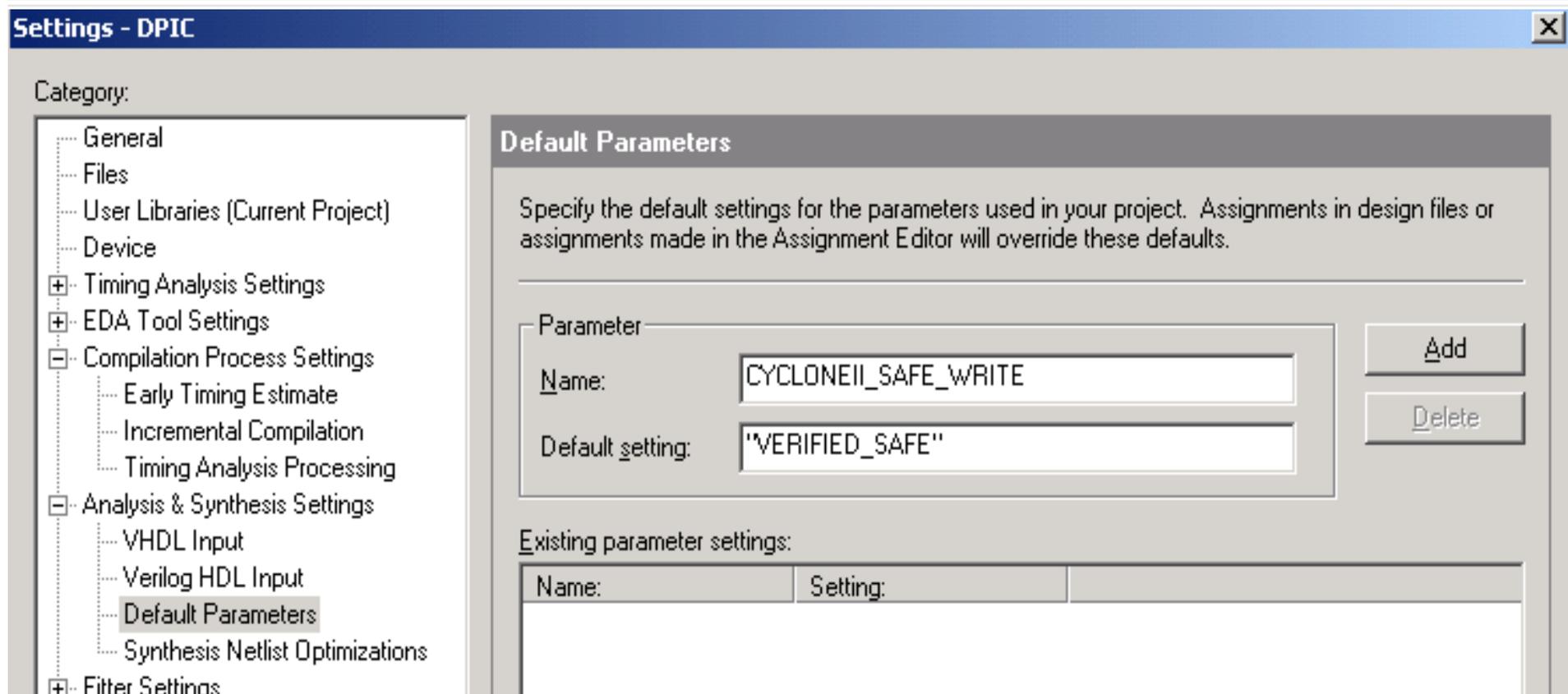


图3-26 键入默认参数

3.2 逻辑数据采样电路设计

Parameter

Name:

Default setting:

Existing parameter settings:

Name:	Setting:
CYCLONEII_SAFE_W...	"VERIFIED_SAFE"

图3-27加入默认参数

3.2 逻辑数据采样电路设计

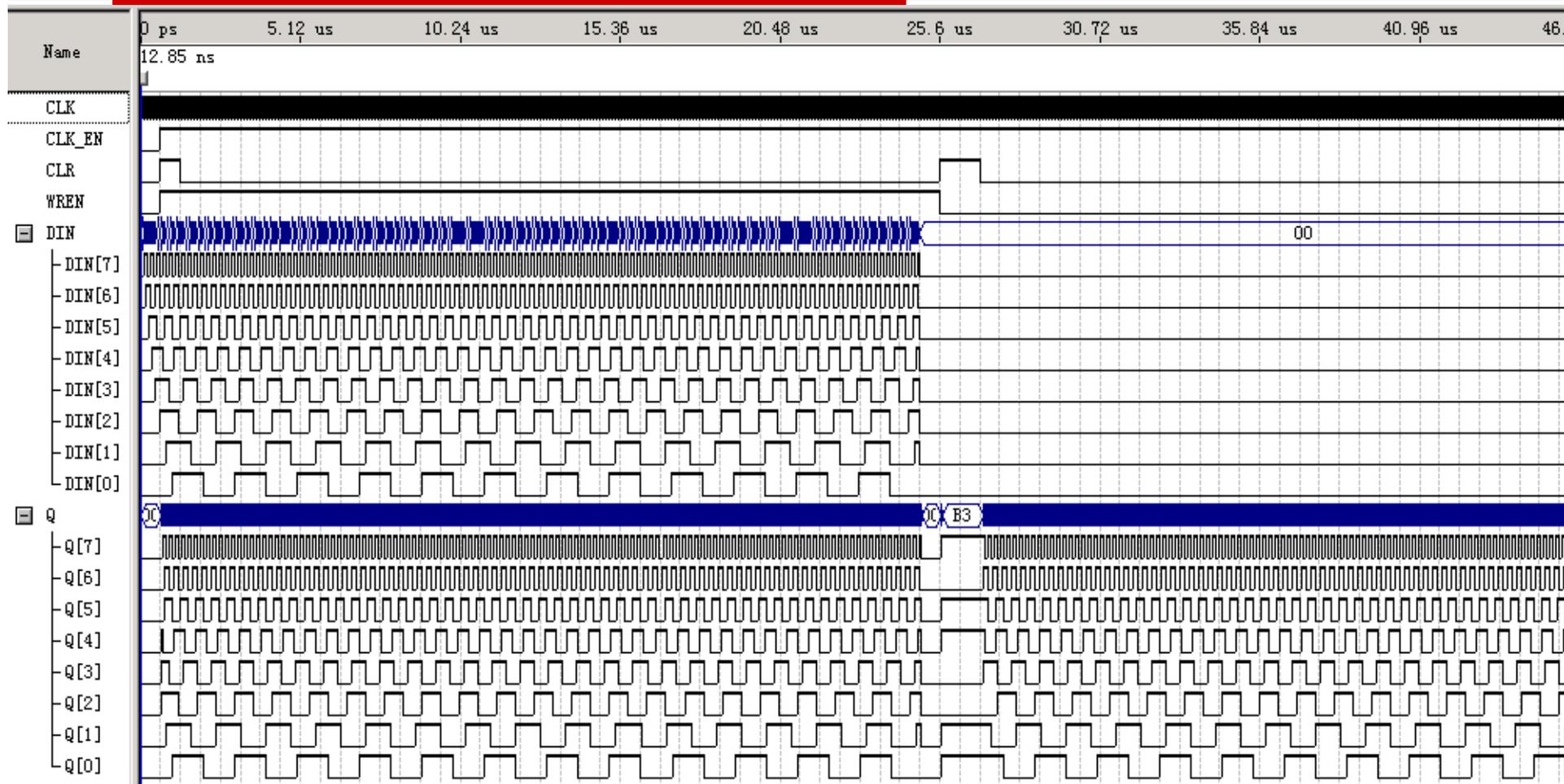
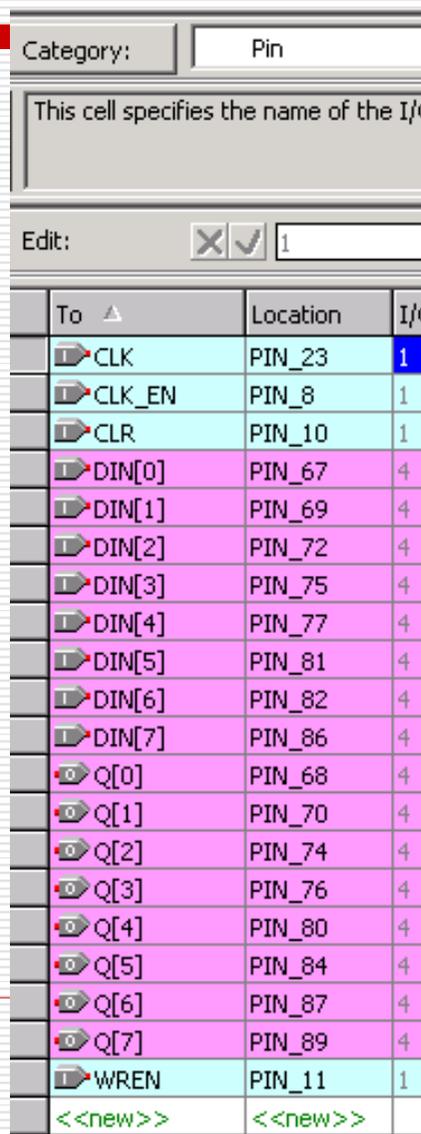


图3-28 逻辑数据采样电路时序仿真波形

3.3 在系统存储器数据读写编辑器应用

1. 锁定引脚



To ▲	Location	I/O
CLK	PIN_23	1
CLK_EN	PIN_8	1
CLR	PIN_10	1
DIN[0]	PIN_67	4
DIN[1]	PIN_69	4
DIN[2]	PIN_72	4
DIN[3]	PIN_75	4
DIN[4]	PIN_77	4
DIN[5]	PIN_81	4
DIN[6]	PIN_82	4
DIN[7]	PIN_86	4
Q[0]	PIN_68	4
Q[1]	PIN_70	4
Q[2]	PIN_74	4
Q[3]	PIN_76	4
Q[4]	PIN_80	4
Q[5]	PIN_84	4
Q[6]	PIN_87	4
Q[7]	PIN_89	4
WREN	PIN_11	1
<<new>>	<<new>>	

图3-29 引脚锁定

3.3 在系统存储器数据读写编辑器应用

2. 打开在系统存储单元编辑窗

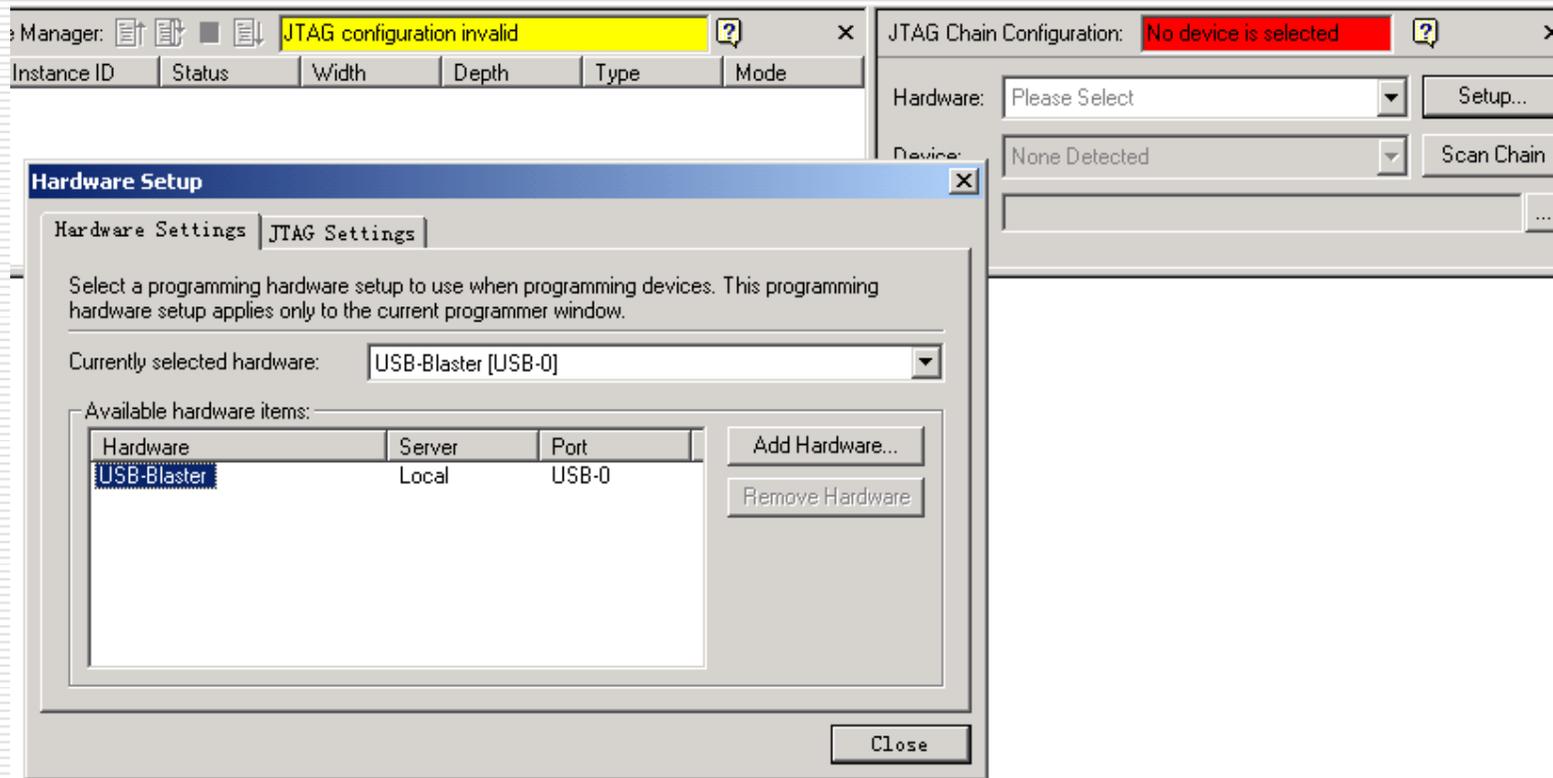


图3-30 In-System Memory Content Editor编辑窗中硬件通信口设置

3.3 在系统存储器数据读写编辑器应用

2. 打开在系统存储单元编辑窗

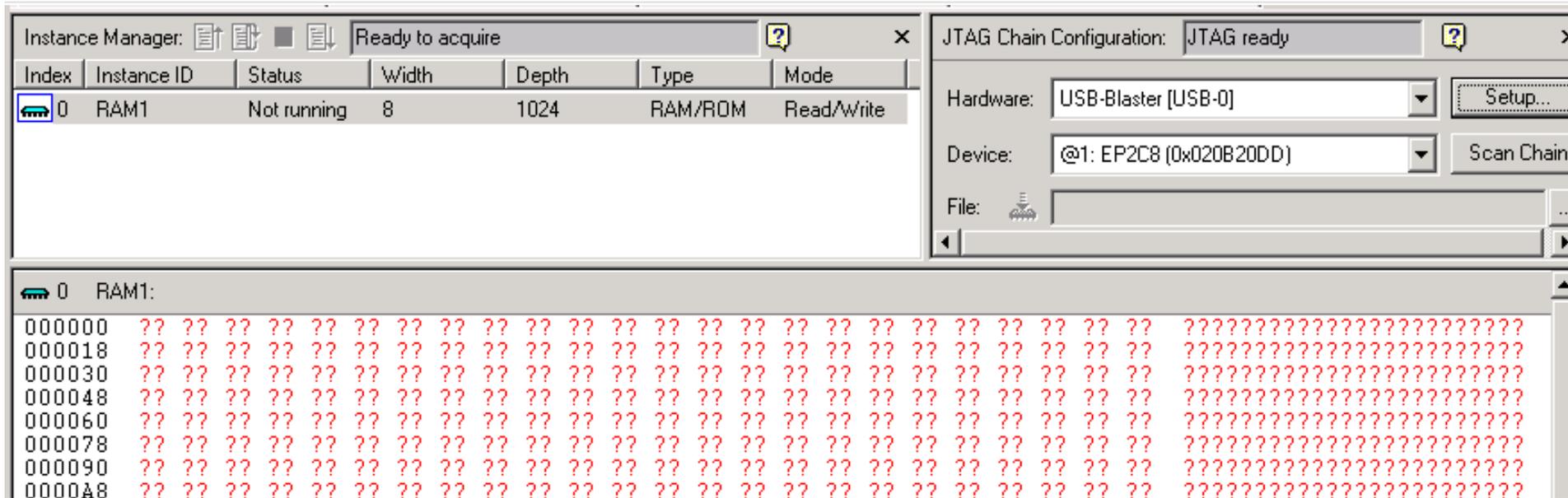


图3-31 In-System Memory Content Editor扫描FPGA结果

3.3 在系统存储器数据读写编辑器应用

3. 读取RAM中的数据

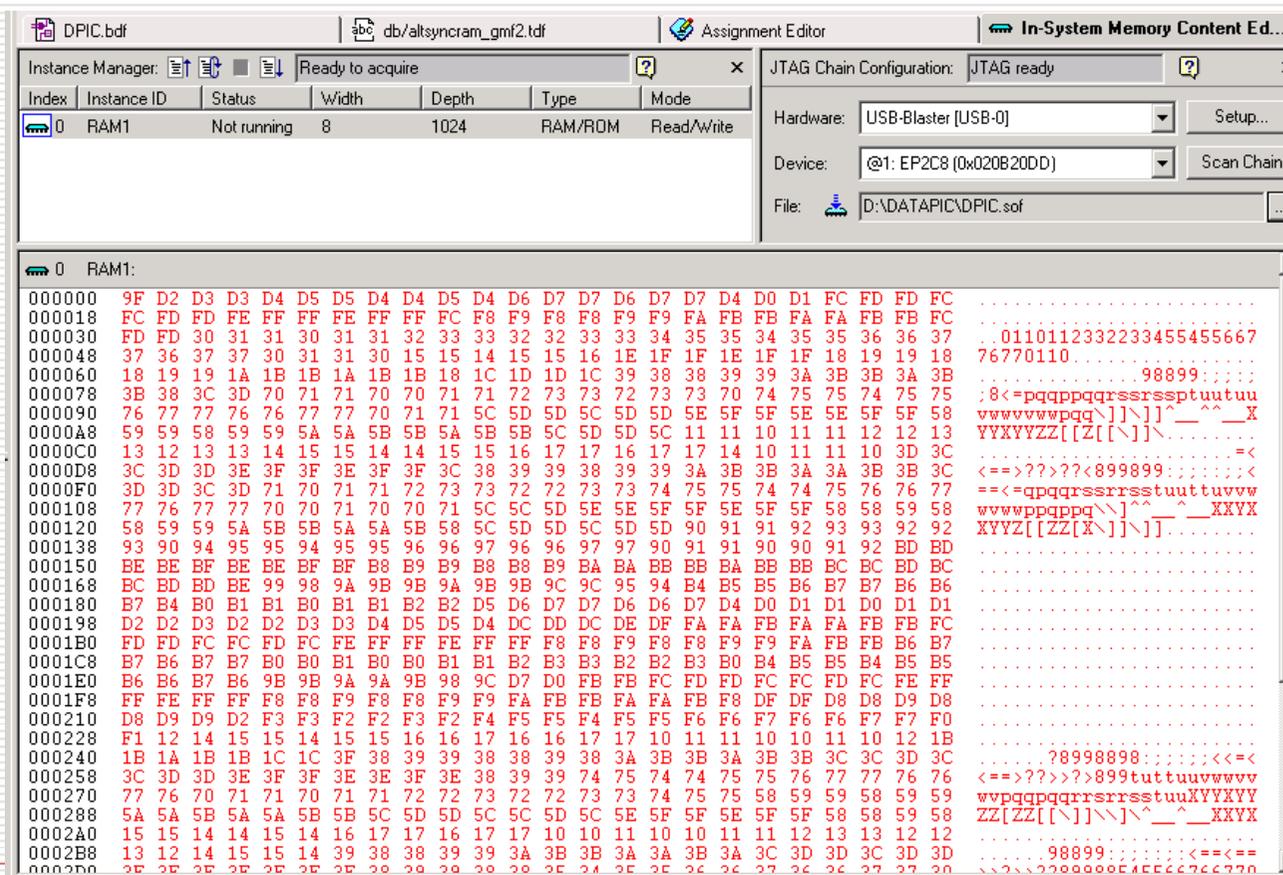


图3-32 In-System Memory Content Editor上载FPGA中RAM数据

3.3 在系统存储器数据读写编辑器应用

3. 读取RAM中的数据

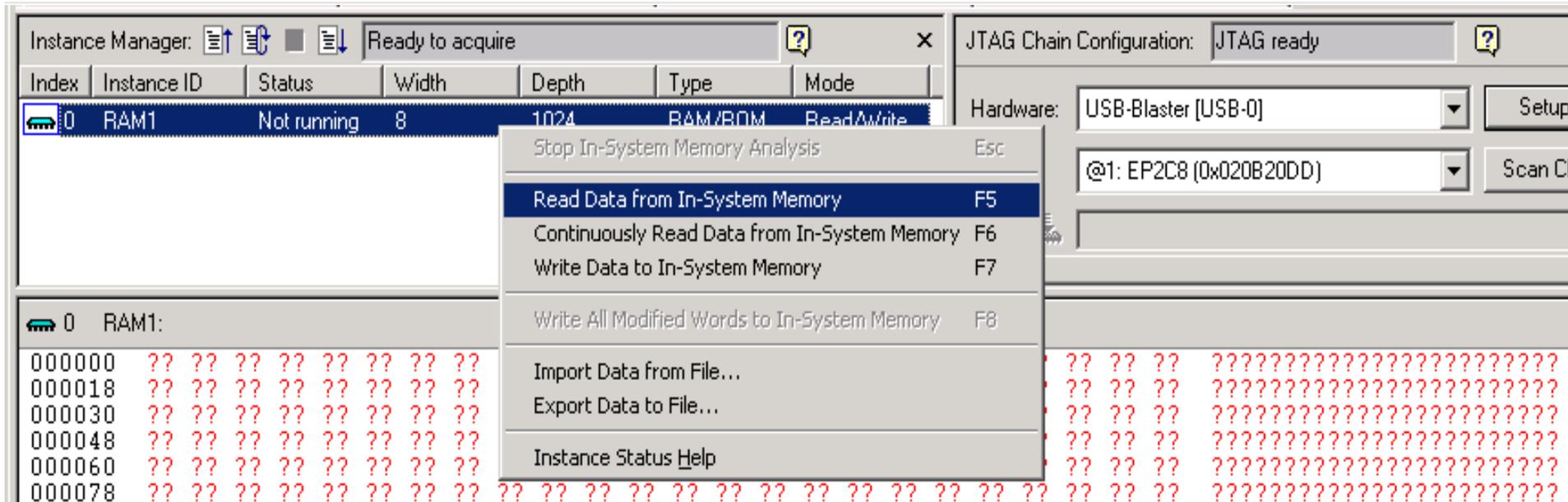


图3-33 利用In-System Memory Content Editor读取LPM_RAM中数据

3.3 在系统存储器数据读写编辑器应用

4. 编辑下载RAM中的数据

5. 输入输出数据文件

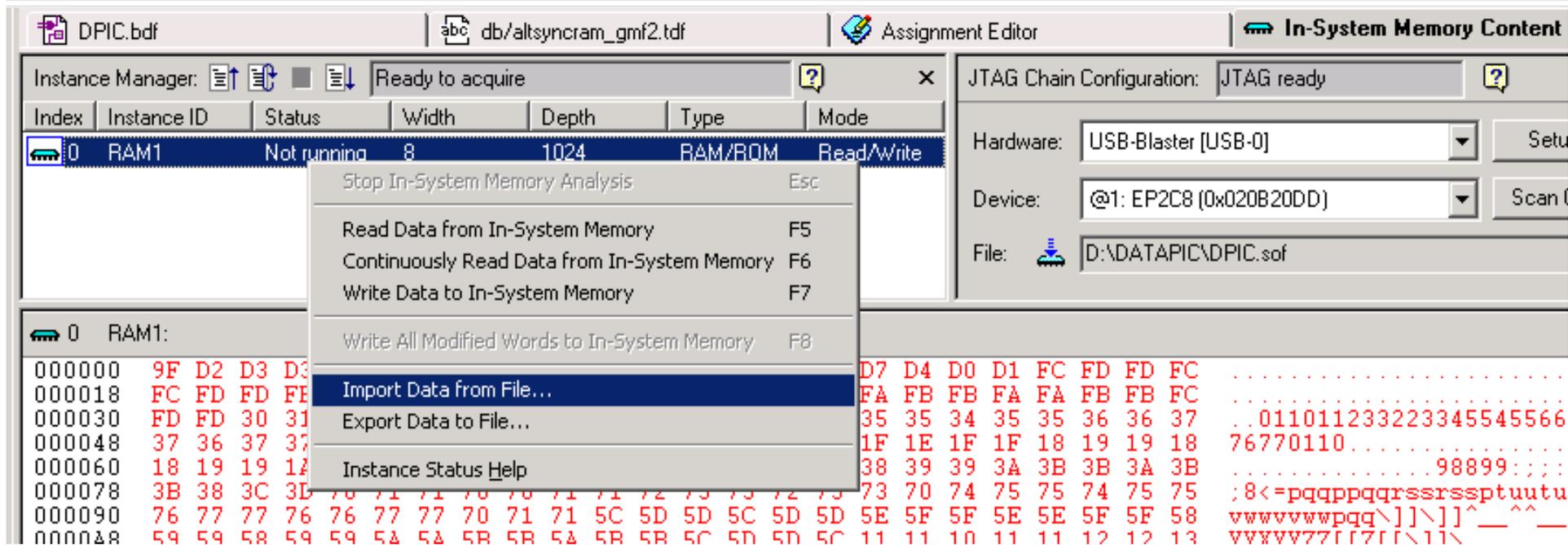


图3-34利用In-System Memory Content Editor向LPM_RAM下载数据文件

3.4 简易正弦信号发生器设计

3.4.1 工作原理

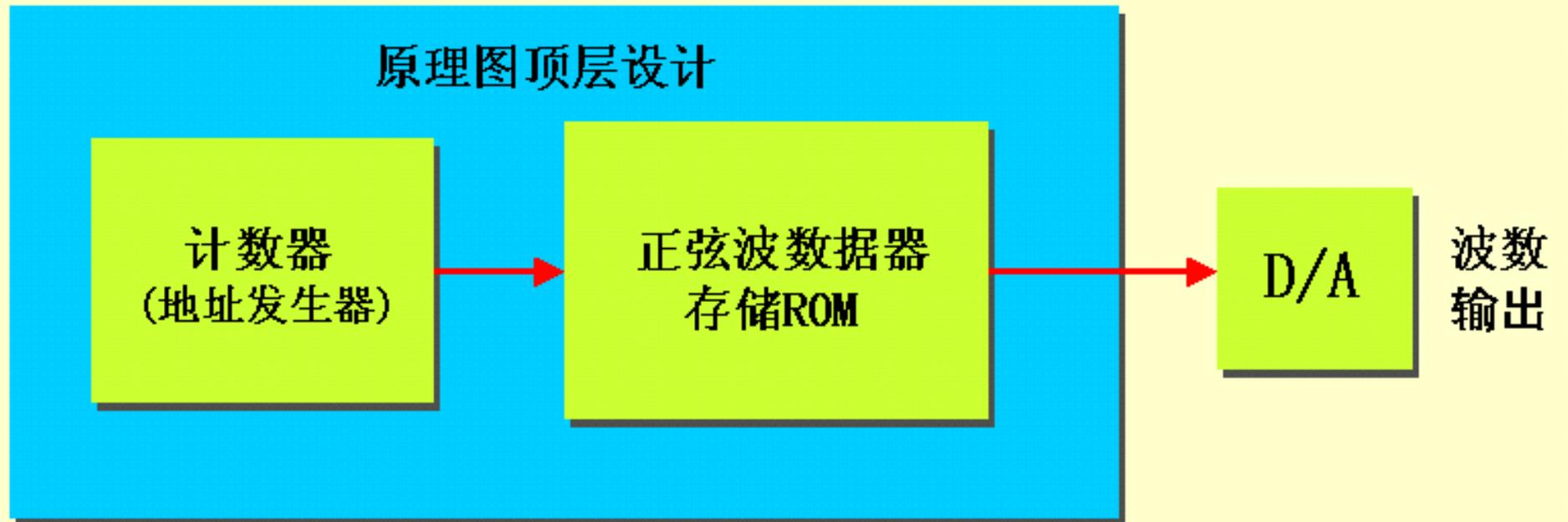


图3-35 正弦信号发生器结构框图

3.4 简易正弦信号发生器设计

3.4.2 定制初始化数据文件

1. 建立.mif格式文件

【例3-1】

```
WIDTH = 8;  
DEPTH = 64;  
ADDRESS_RADIX = HEX;  
DATA_RADIX = HEX;  
CONTENT BEGIN  
    0      :    FF;  
    1      :    FE;  
    2      :    FC;  
    3      :    F9;  
    4      :    F5;  
    ... (数据略去)  
    3D     :    FC;  
    3E     :    FE;  
    3F     :    FF;  
END;
```

3.4 简易正弦信号发生器设计

3.4.2 定制初始化数据文件

1. 建立.mif格式文件

【例3-2】

```
#include <stdio.h>
#include "math.h"
main()
{int i;float s;
for (i=0;i<1024; i++)
    { s = sin(atan(1)*8*i/1024);
    printf ("%d : %d;\n",i,(int)((s+1)*1023/2));
    }
}
```

把上述程序编译成程序后，可在DOS命令行下执行命令：
romgen > sin_ rom. mif;

3.4 简易正弦信号发生器设计

3.4.2 定制初始化数据文件

2. 建立.hex格式文件

Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	255	254	252	249	245	239	233	225
8	217	207	197	186	174	162	150	137
16	124	112	99	87	75	64	53	43
24	34	26	19	13	8	4	1	0
32	0	1	4	8	13	19	26	34
40	43	53	64	75	87	99	112	124
48	137	150	162	174	186	197	207	217
56	225	233	239	245	249	252	254	255

图3-36 将波形数据填入mif文件表中

3.4 简易正弦信号发生器设计

3.4.2 定制初始化数据文件

2. 建立.hex格式文件



图3-37 ASM格式建hex文件

3.4 简易正弦信号发生器设计

3.4.2 定制初始化数据文件

2. 建立.hex格式文件

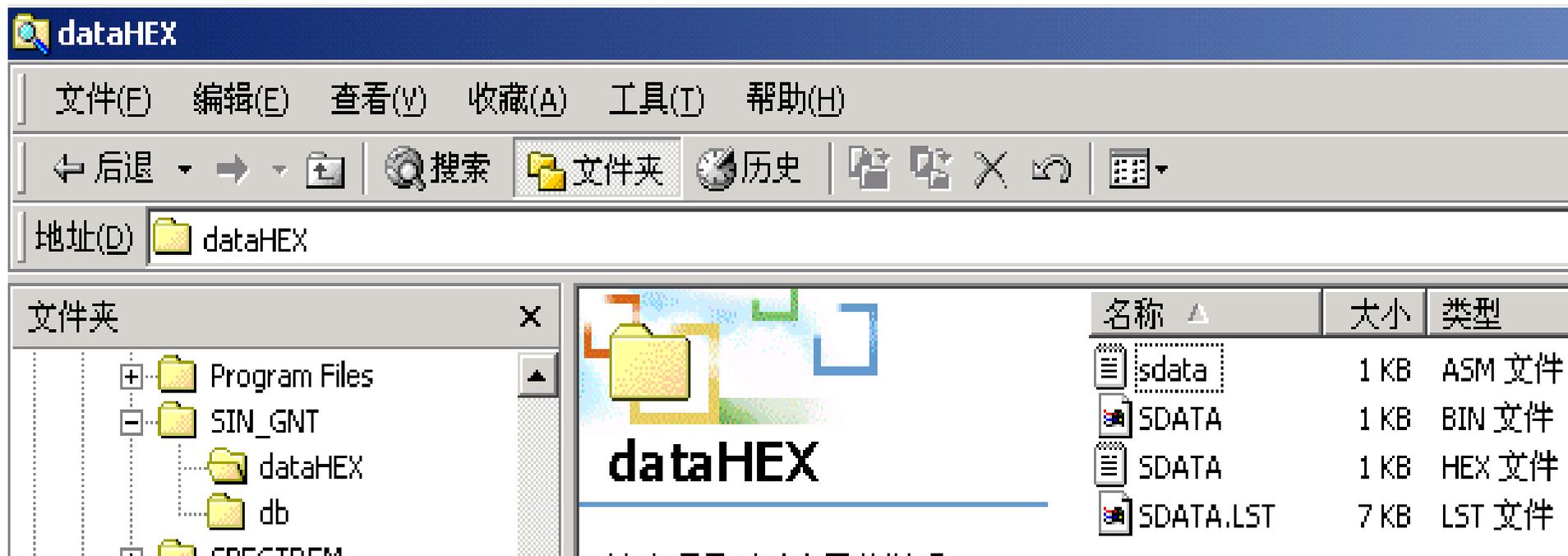


图3-38 sdata.hex文件的放置路径

3.4 简易正弦信号发生器设计

3.4.3 定制LPM元件

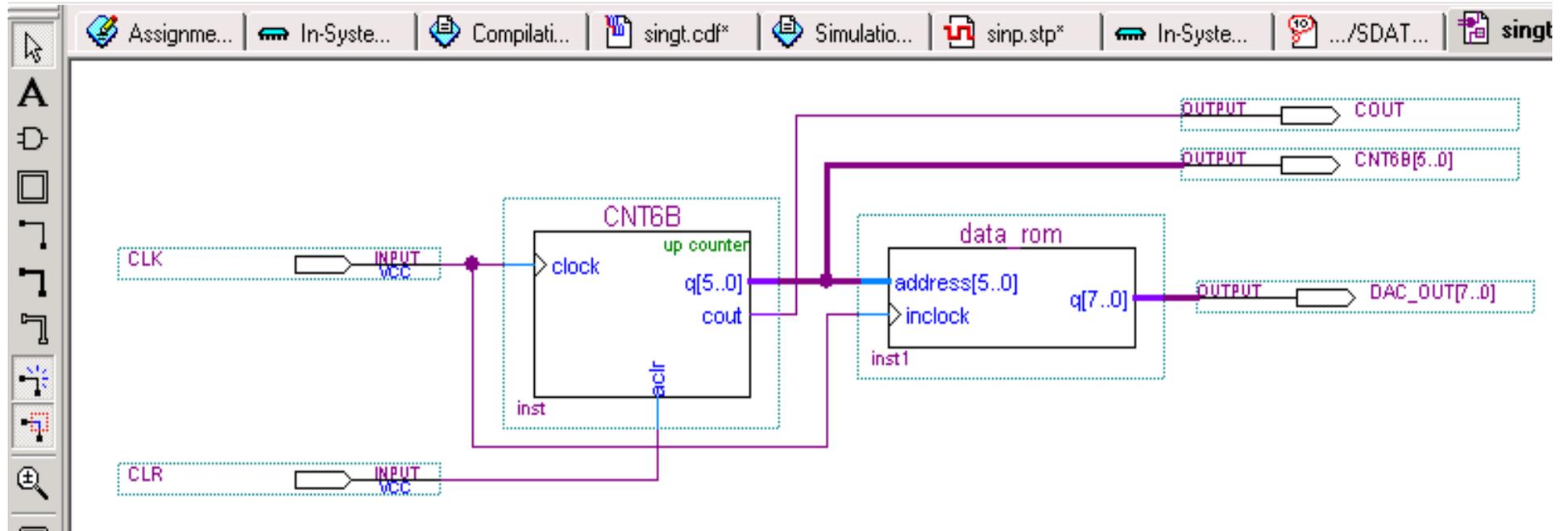


图3-39 简易正弦信号发生器顶层电路设计

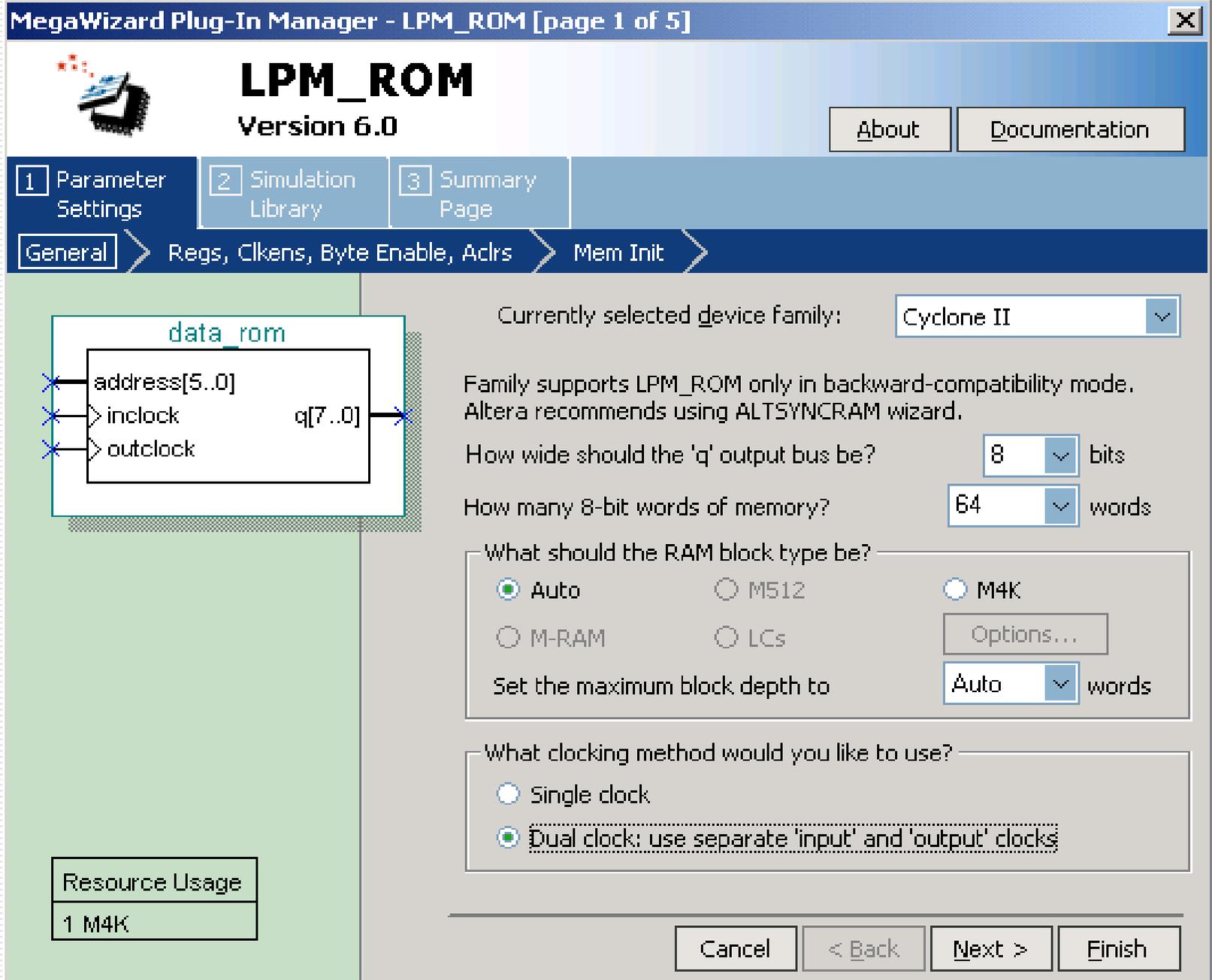


图3-40 选择data_rom模块数据线和地址线宽度

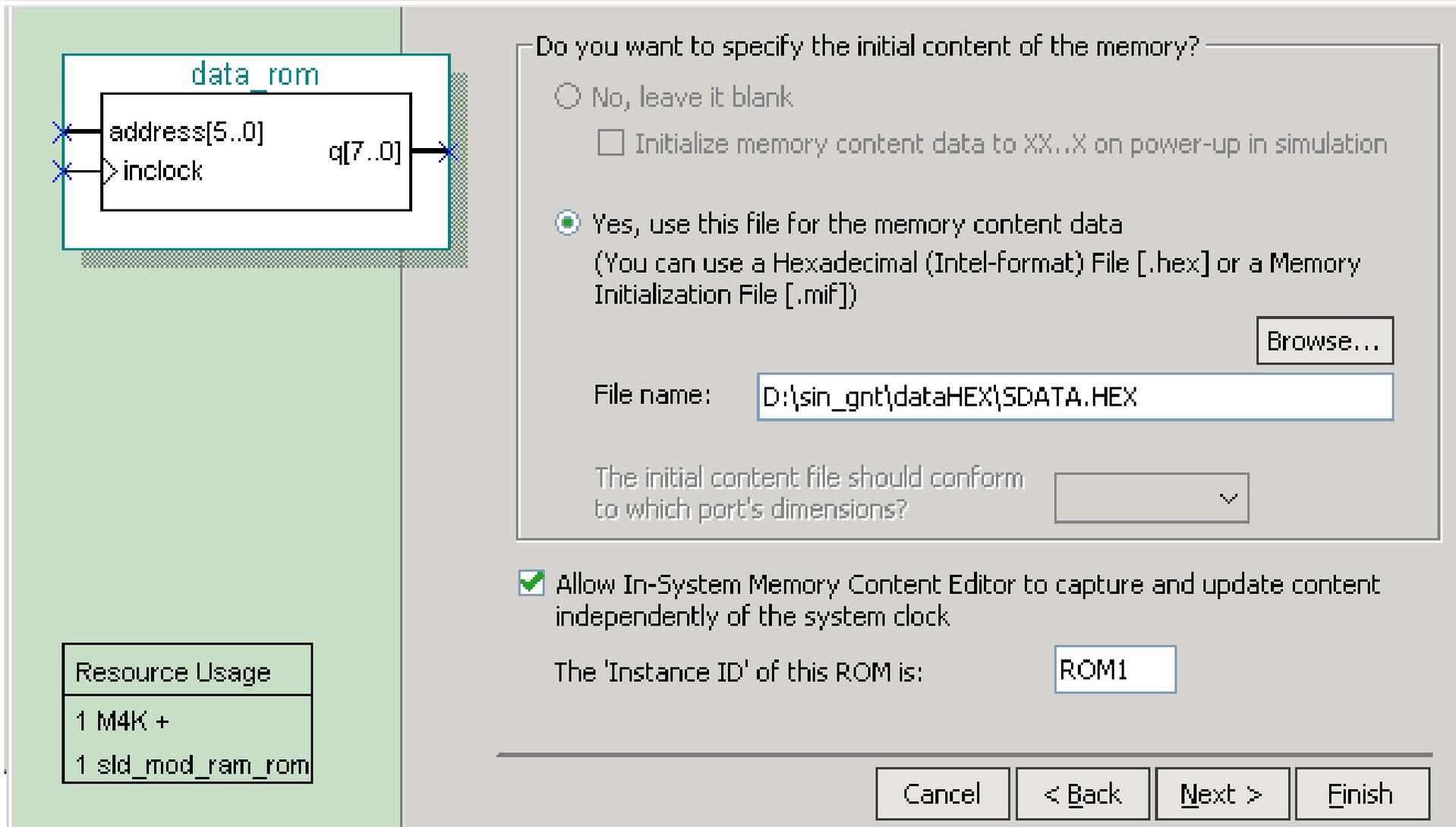


图3-41 调入ROM初始化数据文件并选择在系统读写功能

3.4 简易正弦信号发生器设计

3.4.3 定制LPM元件

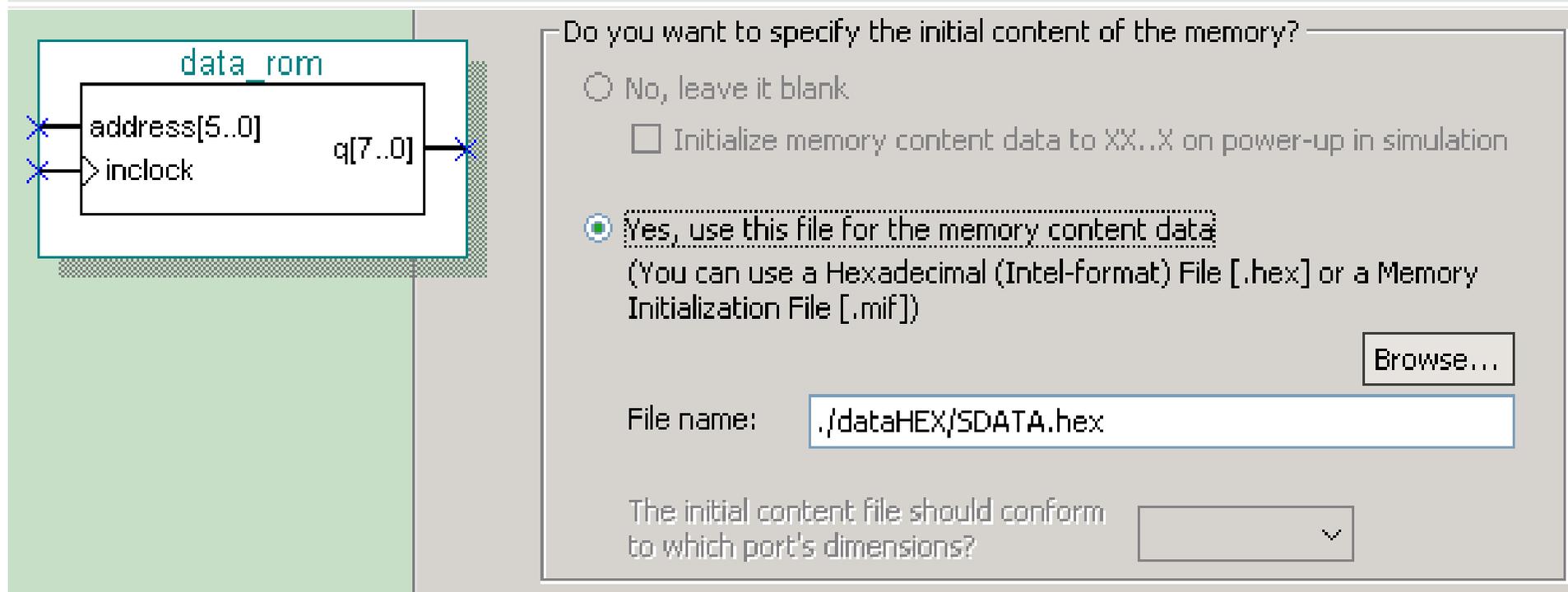


图3-42 修改初始化数据文件路径

3.4 简易正弦信号发生器设计

3.4.3 定制LPM元件

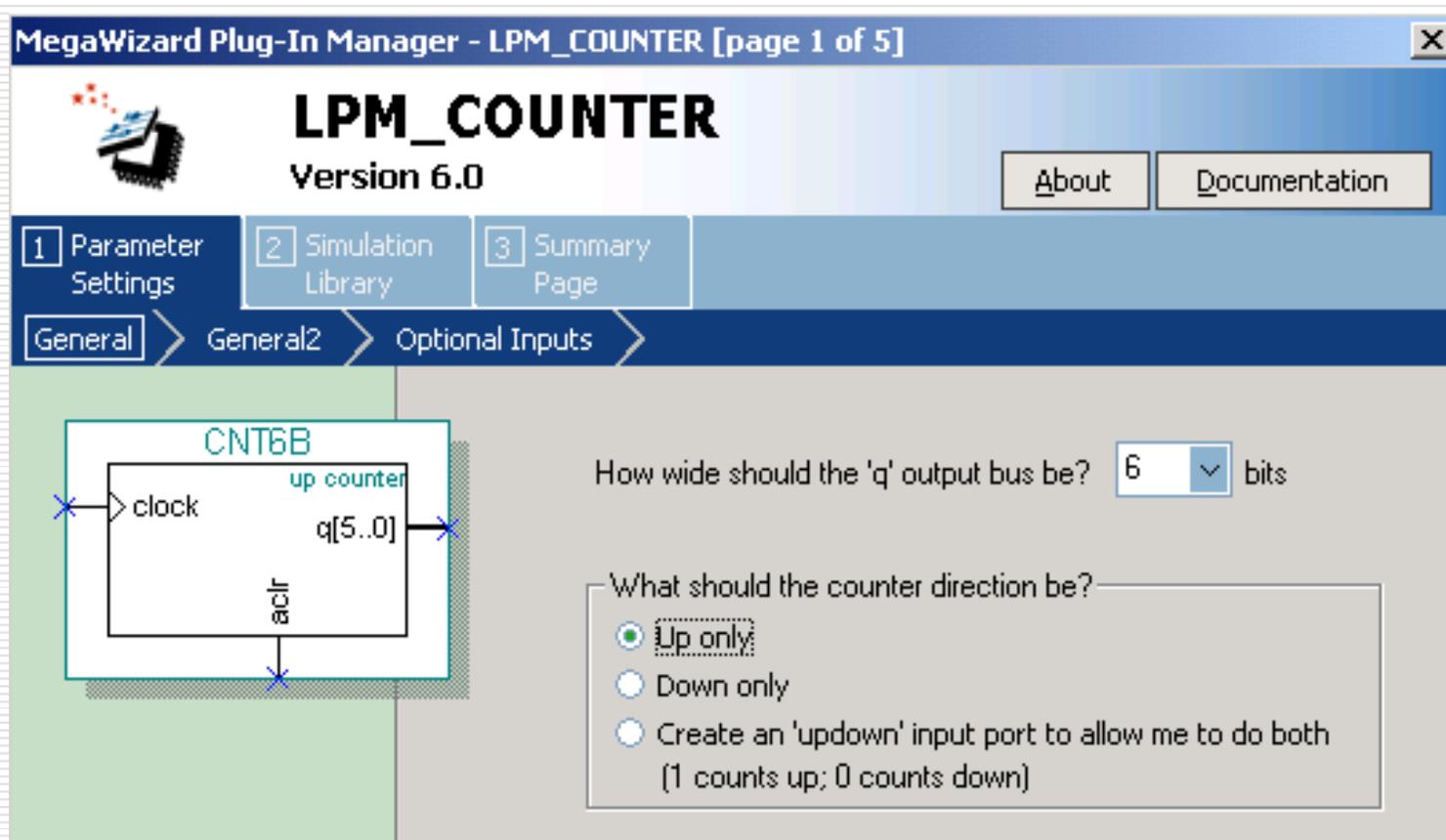


图3-43 设定为加法计数器

3.4 简易正弦信号发生器设计

3.4.4 完成顶层设计

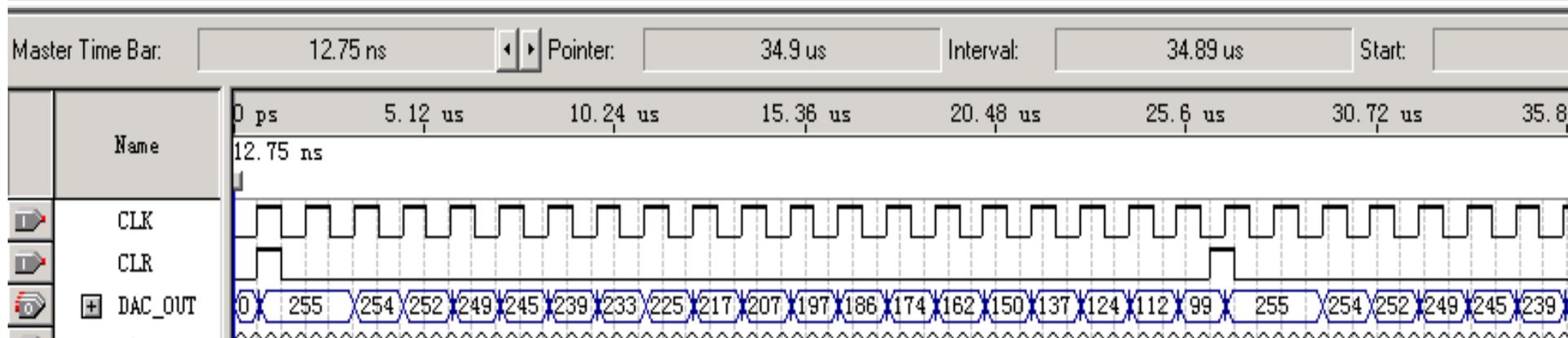


图3-44 当前工程仿真波形输出

3.4 简易正弦信号发生器设计

3.4.4 完成顶层设计

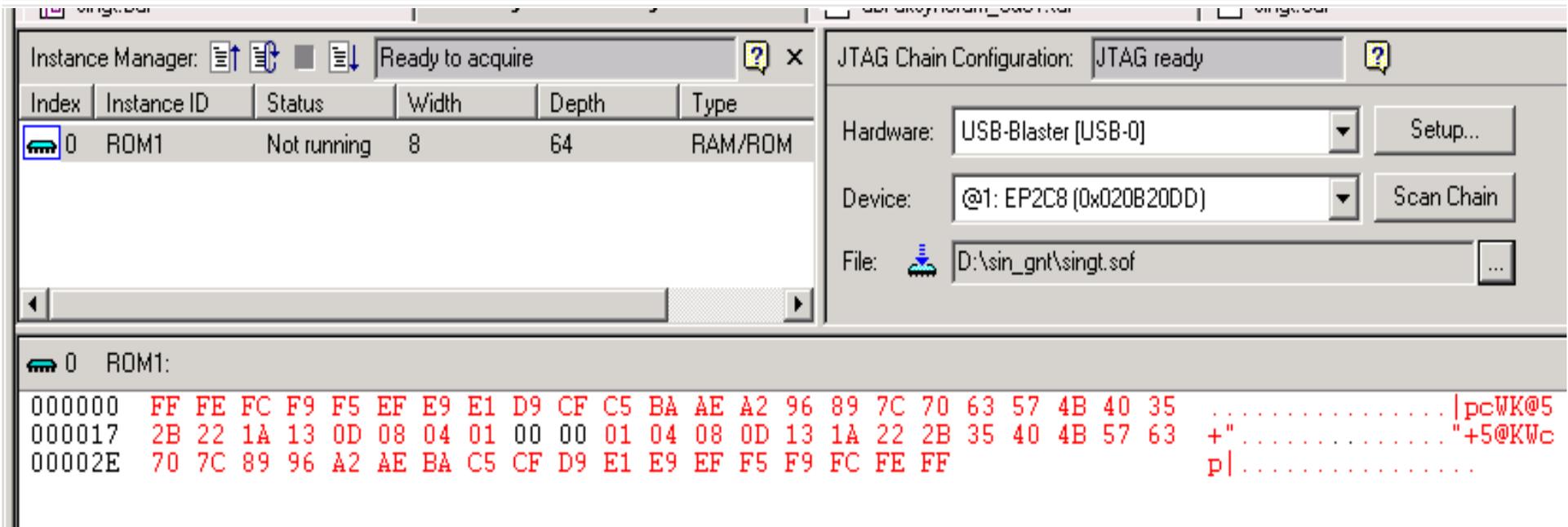


图3-45利用In-System Memory Content Editor读取LPM_ROM中数据

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II 一般使用方法和实例

1. 打开SignalTap II 编辑窗

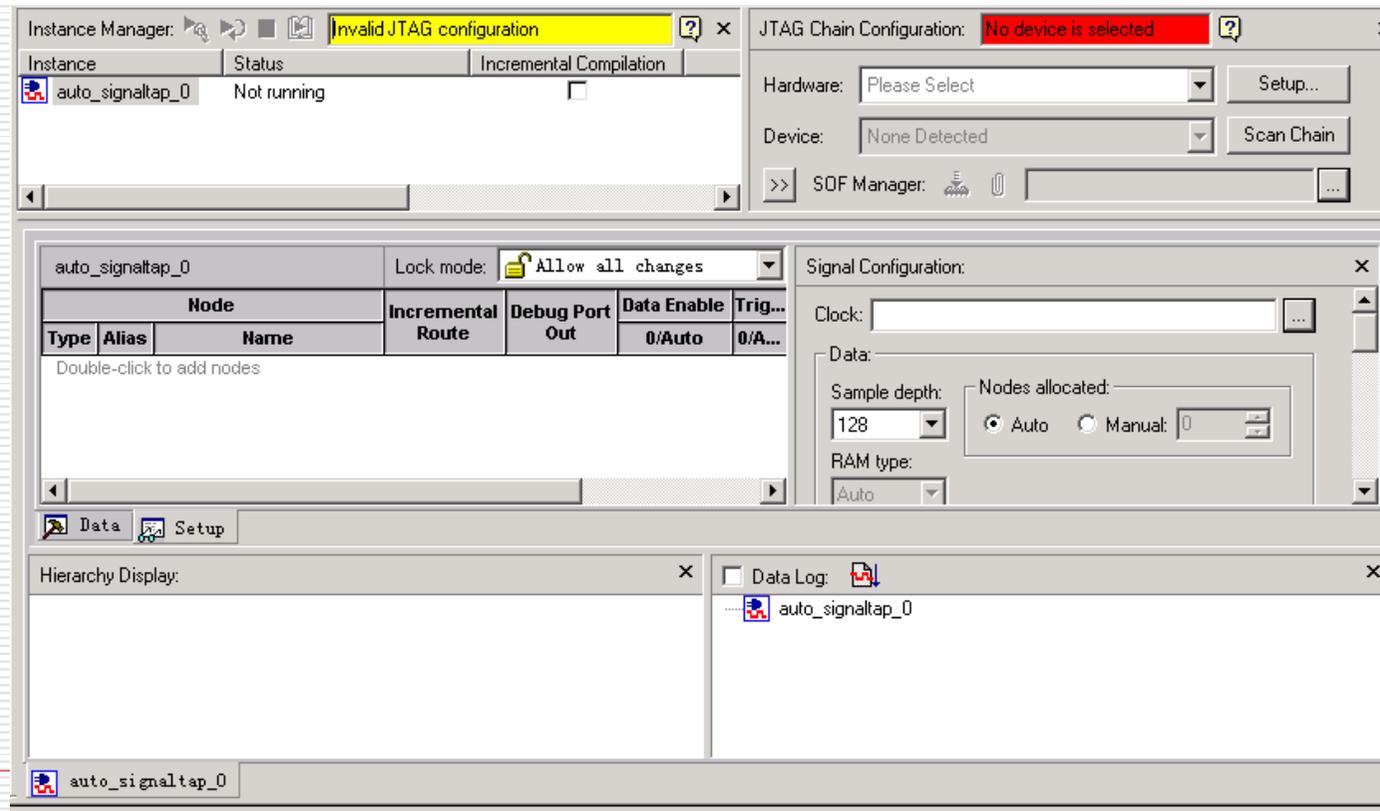


图3-46 SignalTap II 编辑窗

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II一般使用方法和实例

2. 调入待测信号窗

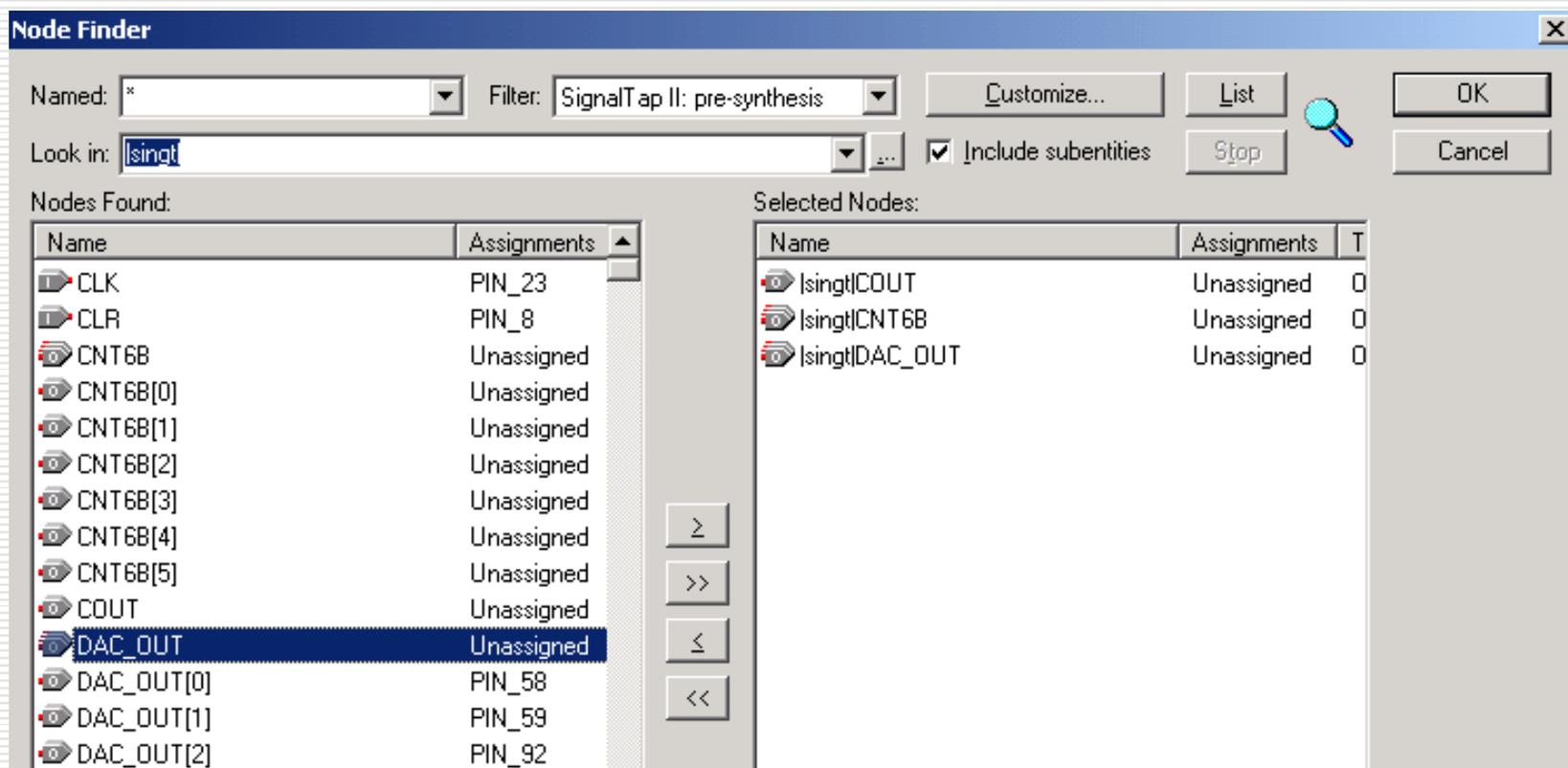


图3-47 选择需要测试的信号

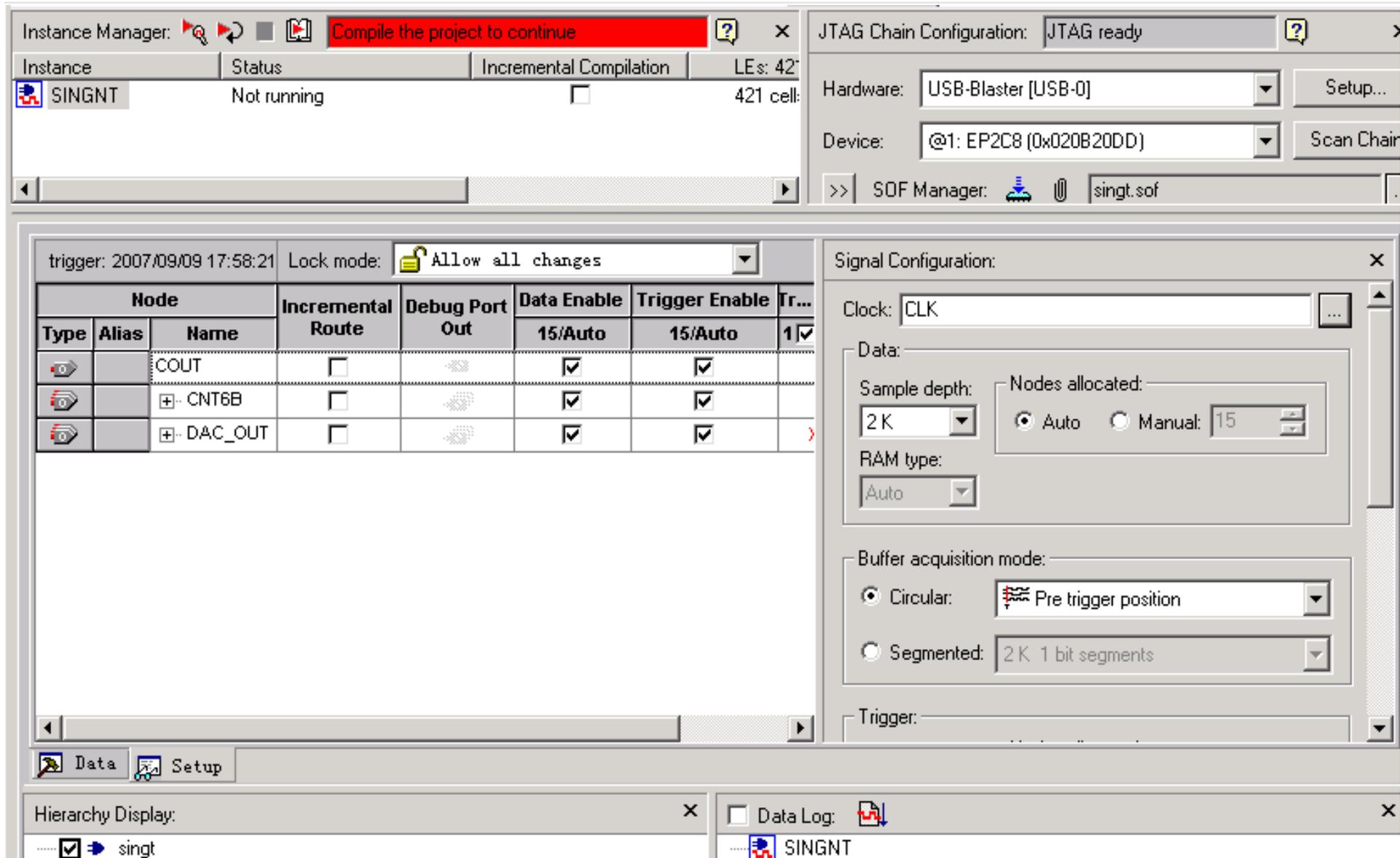


图3-48 设置SignalTap II工作参数

3. SignalTap II 参数设置

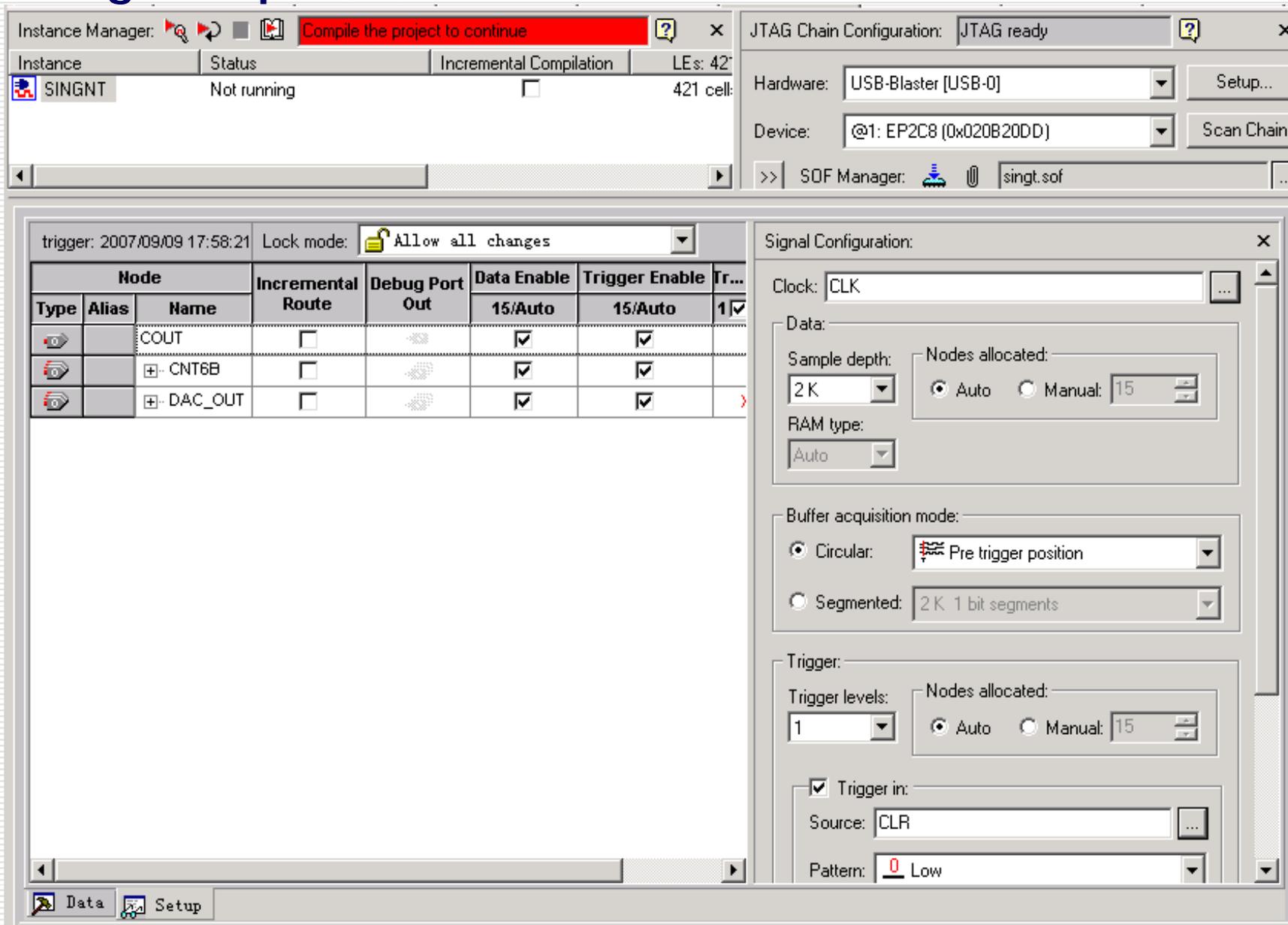


图3-49 设置SignalTap II的触发信号和触发方式

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II 一般使用方法和实例

4. 文件存盘

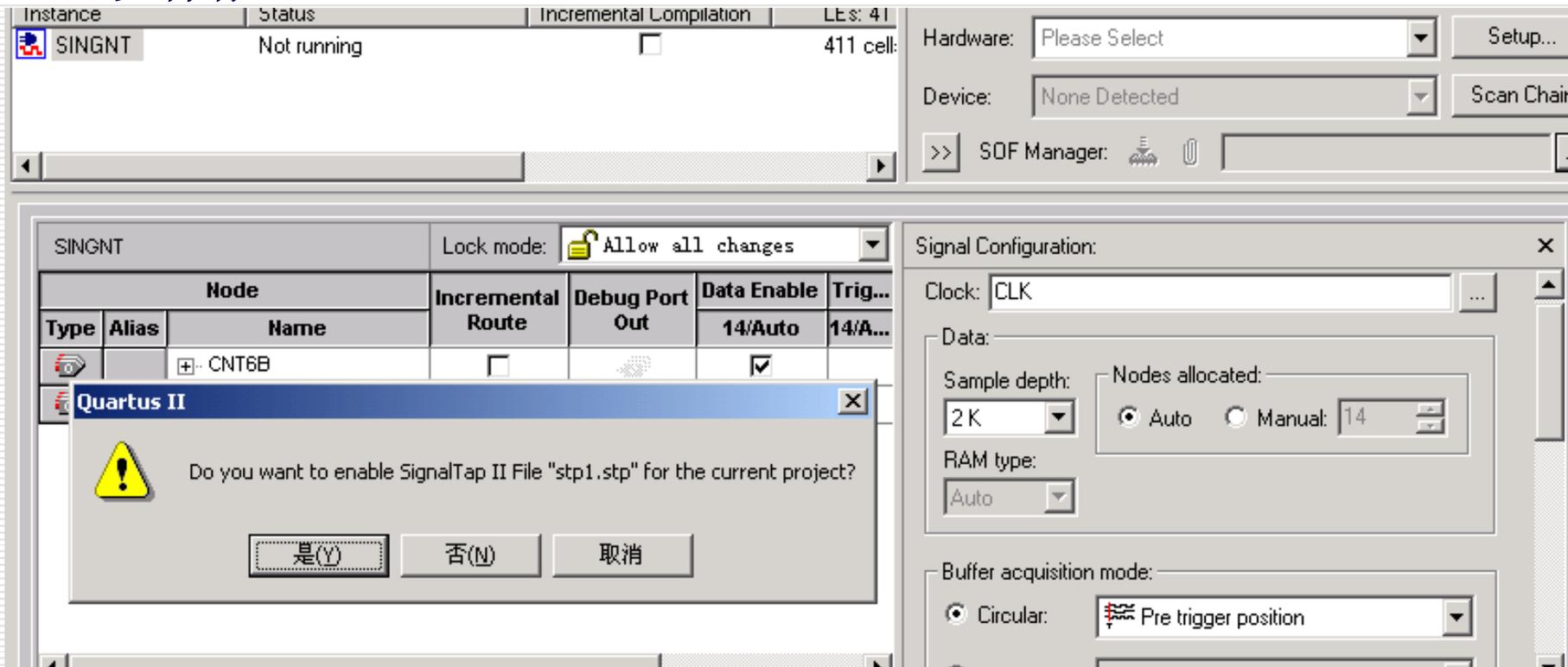


图3-50 SignalTap II 文件存盘

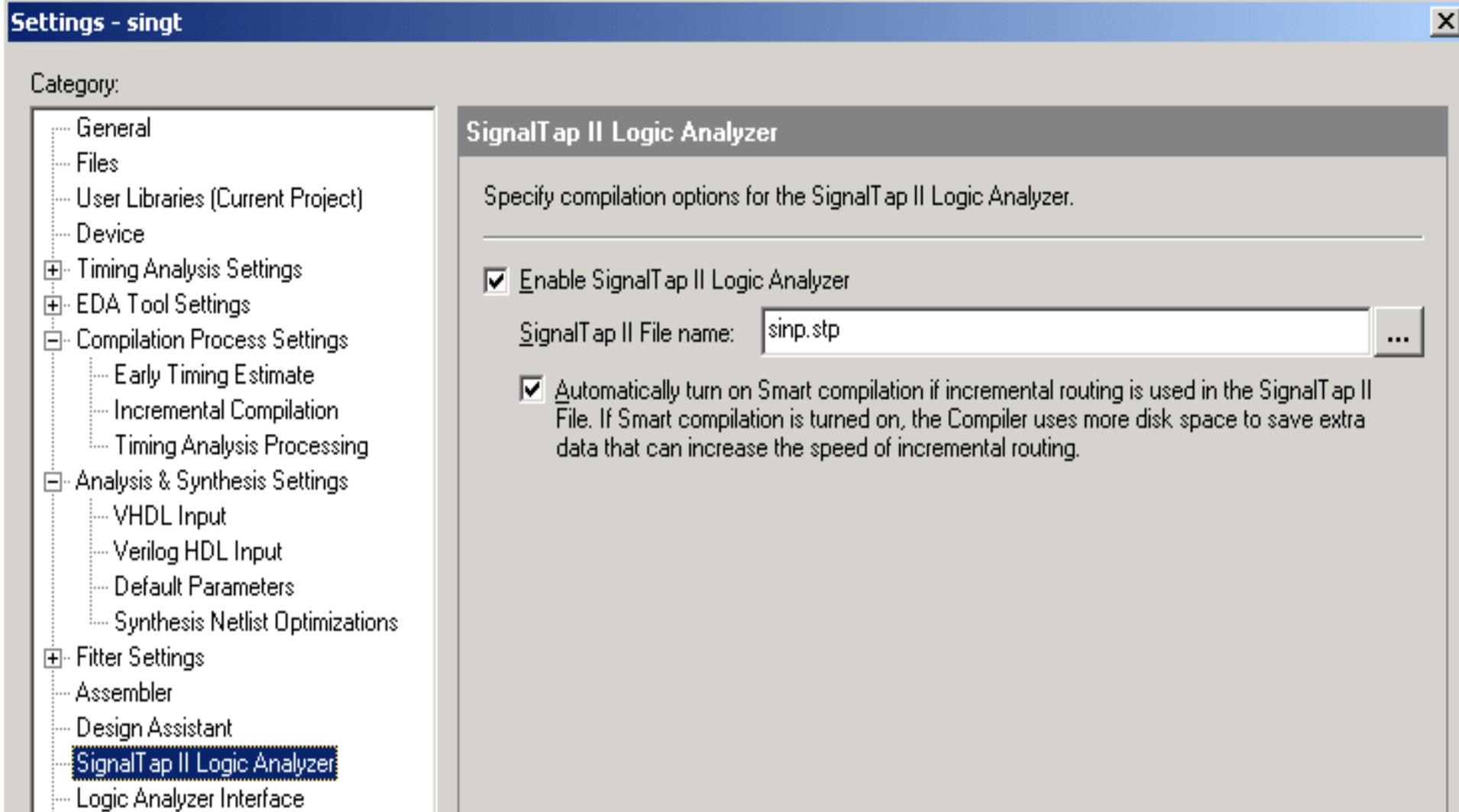


图3-51 设置全程编译中加入SignalTap II核文件

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II一般使用方法和实例

5. 编译下载

6. 启动SignalTap II进行采样与分析

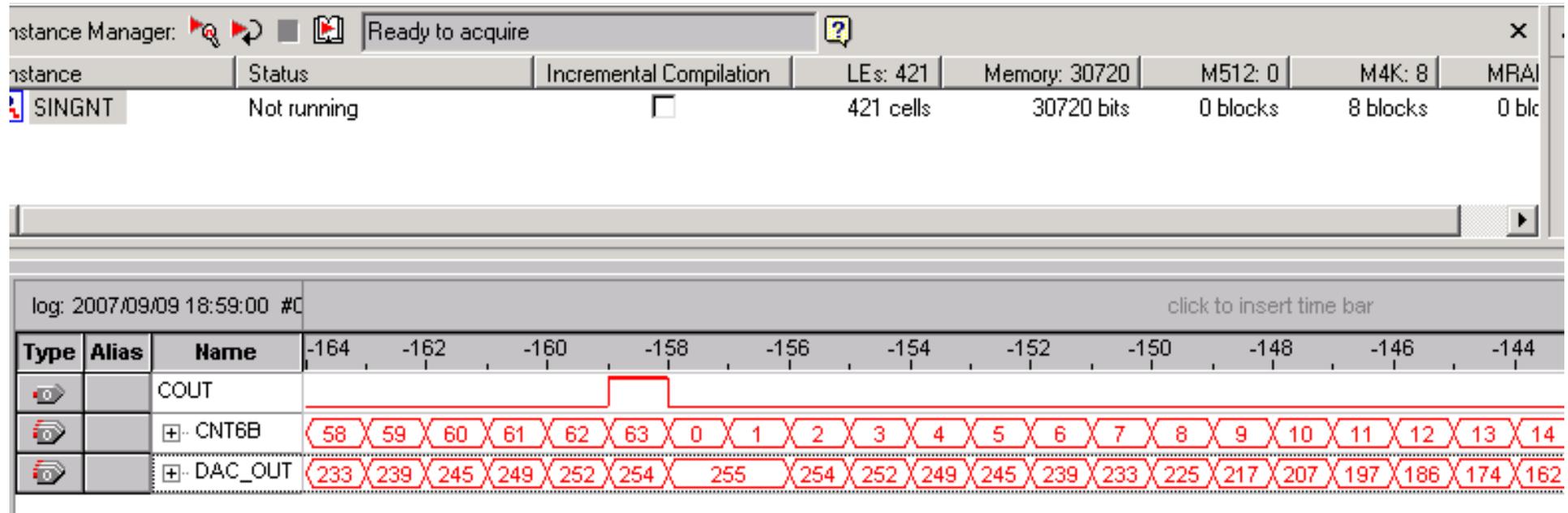


图3-52 下载含有SignalTap II的.sof文件并启动SignalTap II

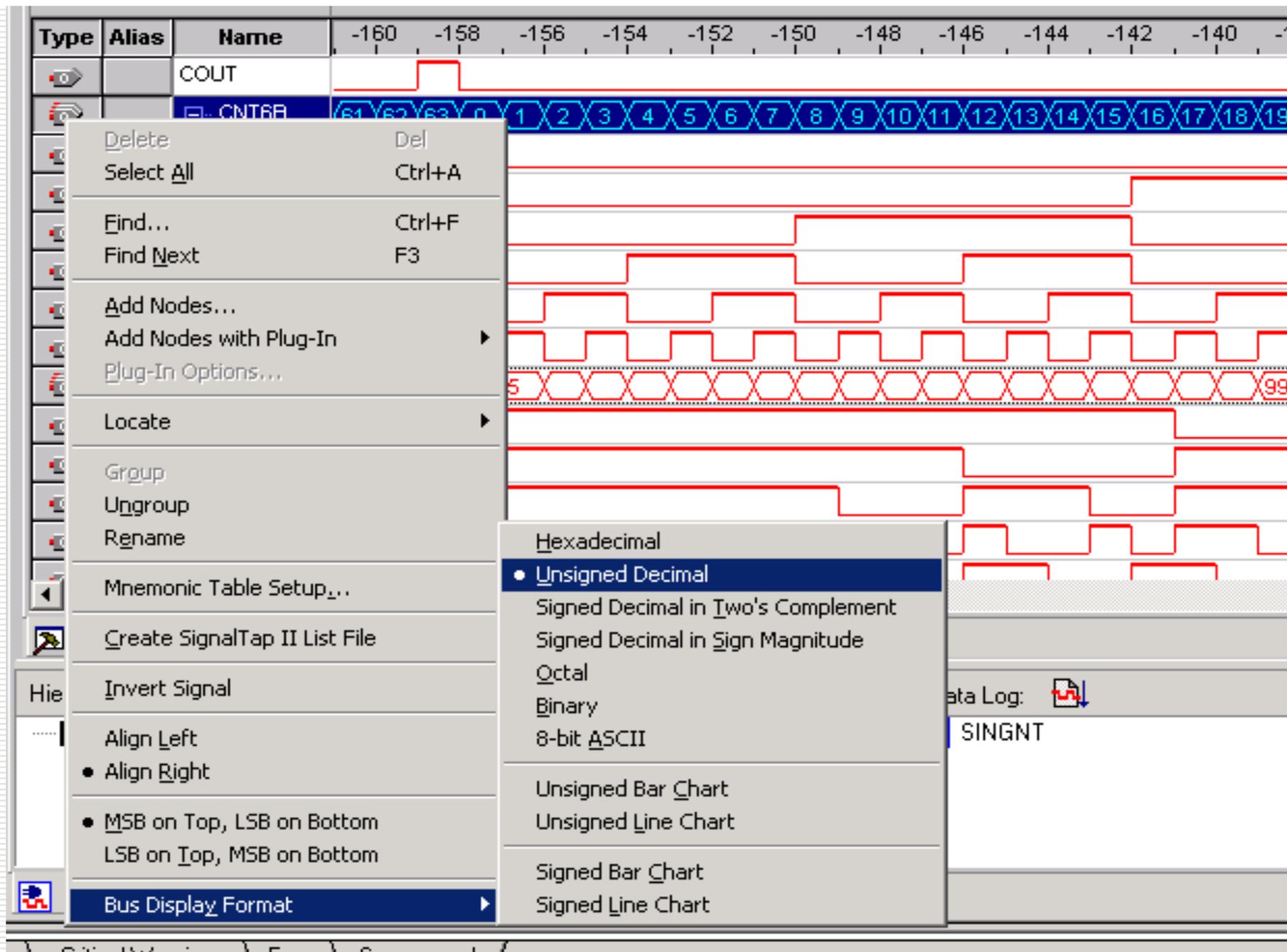


图3-53 设置SignalTap II窗口中波形数据显示方式

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II 一般使用方法和实例

6. 启动SignalTap II进行采样与分析

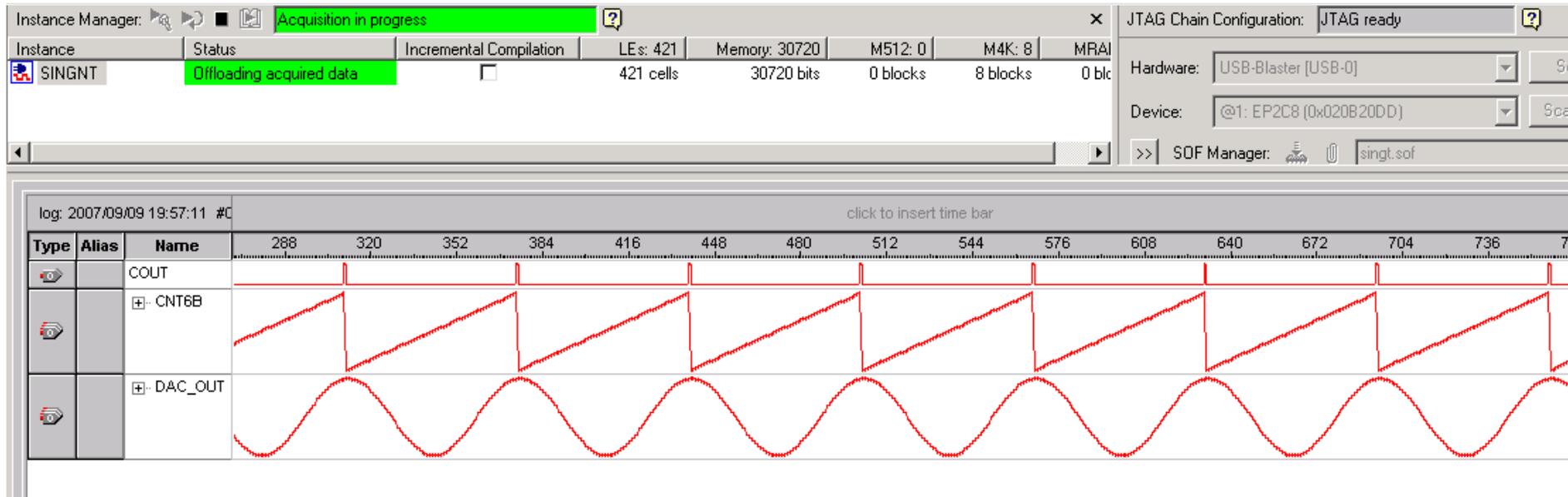


图3-54 SignalTap II 嵌入式逻辑分析仪获得的波形

3.5 嵌入式逻辑分析仪使用方法

3.5.1 SignalTap II一般使用方法和实例

6. 启动SignalTap II进行采样与分析

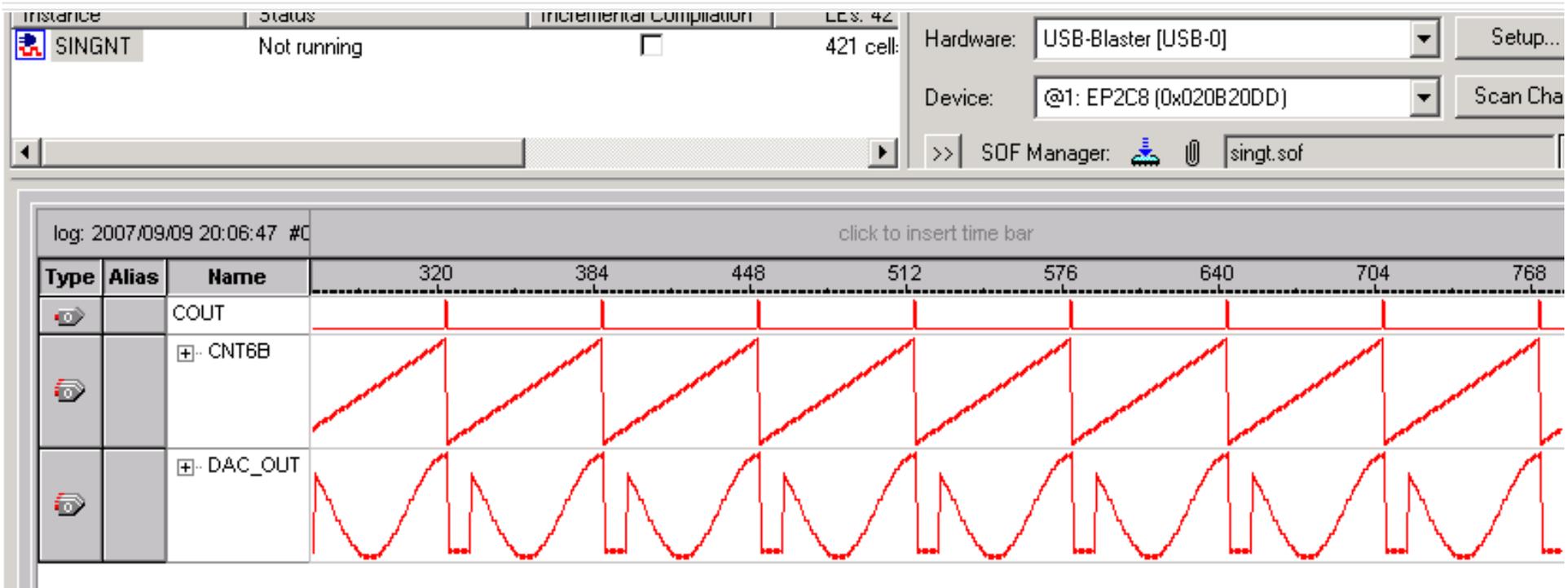


图3-55 利用In-System Memory Content Editor修改LPM_ROM中数据后SignalTap II测得的波形

3.5 嵌入式逻辑分析仪使用方法

3.5.2 编辑SignalTapII的触发信号

The screenshot shows the SignalTap II configuration interface. At the top, it displays 'trigger: 2004/12/03 13:35:38 #1' and a 'Lock mode' dropdown set to 'Allow all changes'. Below this is a table with columns: Node (Type, Alias, Name), Incremental Route, Debug Port Out, Data Enable (14/Auto), Trigger Enable (14/Auto), and Trigger Levels (1, Basic/Advanced). Two nodes are listed: DOUT and Q1. The 'Trigger Levels' dropdown for the first node is open, showing 'Basic' and 'Advanced' options, with 'Advanced' selected. To the right is a 'Signal' panel with 'Clock' set to 'CLK', 'Data' field, 'Sample' rate set to '1 K', and 'Nodes' set to 'Aut'.

Node			Incremental Route	Debug Port Out	Data Enable	Trigger Enable	Trigger Levels
Type	Alias	Name			14/Auto	14/Auto	1 Basic
		+ DOUT	<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	x Basic Advanced
		+ Q1	<input type="checkbox"/>		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	XXXXXXXX

图3-56 选择高级触发条件

3.5 嵌入式逻辑分析仪使用方法

3.5.2 编辑SignalTap II的触发信号

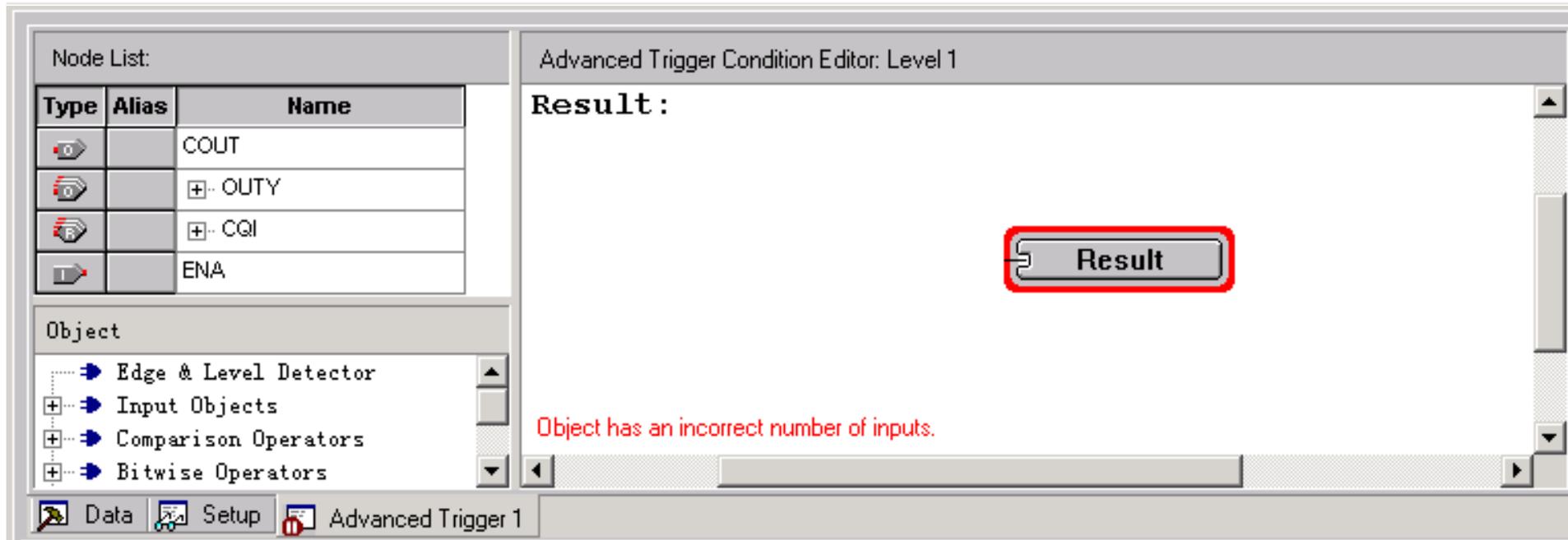


图3-57 进入“触发条件函数编辑”窗口

3.5 嵌入式逻辑分析仪使用方法

3.5.2 编辑SignalTap II的触发信号

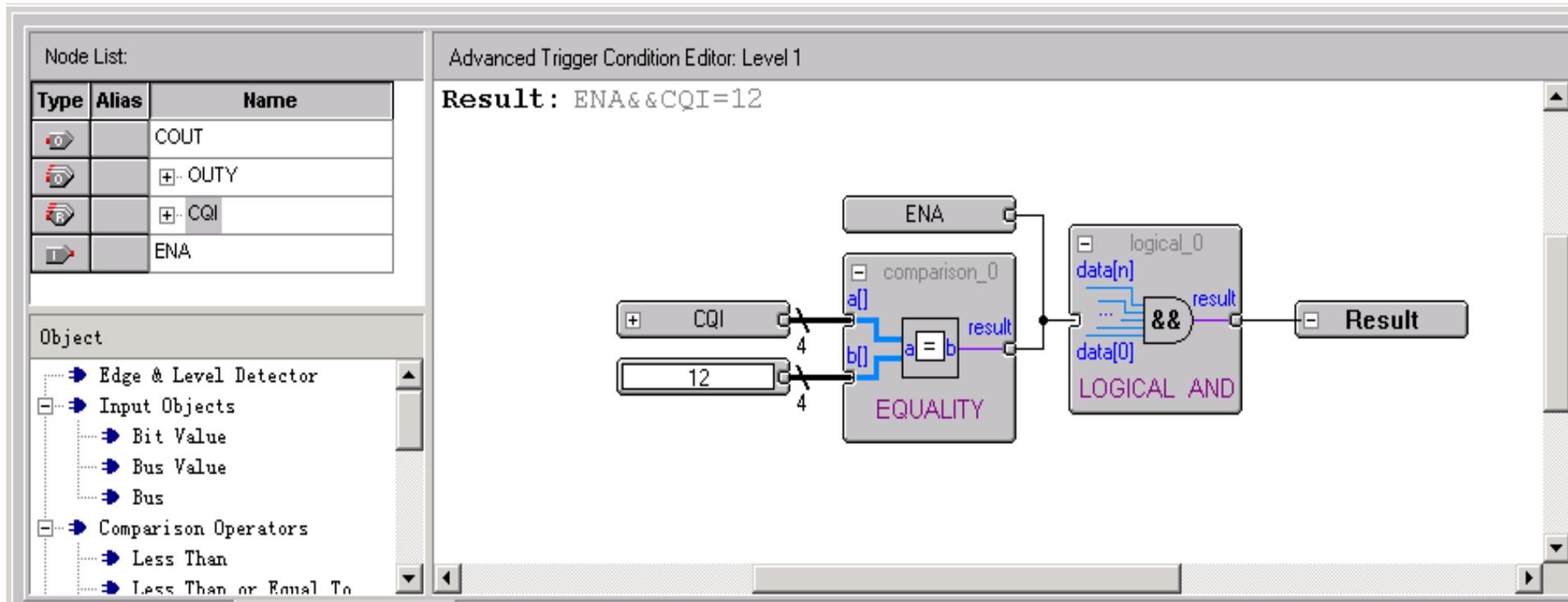
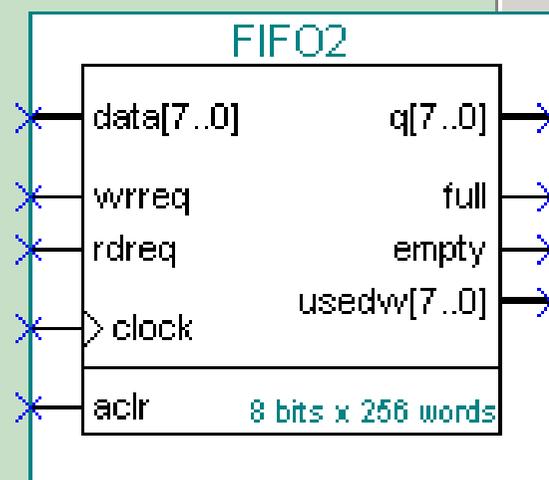


图3-58 编辑触发函数

3.6 FIFO模块定制



Output register option for devices with fully synchronous RAM
Would you like to register the output to maximize performance but use more area?

Yes (best speed)

No (smallest area)

Would you like to disable any circuitry protection?

If not required, overflow and underflow checking can be disabled to improve performance.

Disable overflow checking. Writing to a full FIFO will

Disable underflow checking. Reading from an empty FIFO will corrupt contents.

Implement FIFO function with logic cells only, even if the device contains EABs or ESBs

图3-59 FIFO编辑窗

3.6 FIFO模块定制

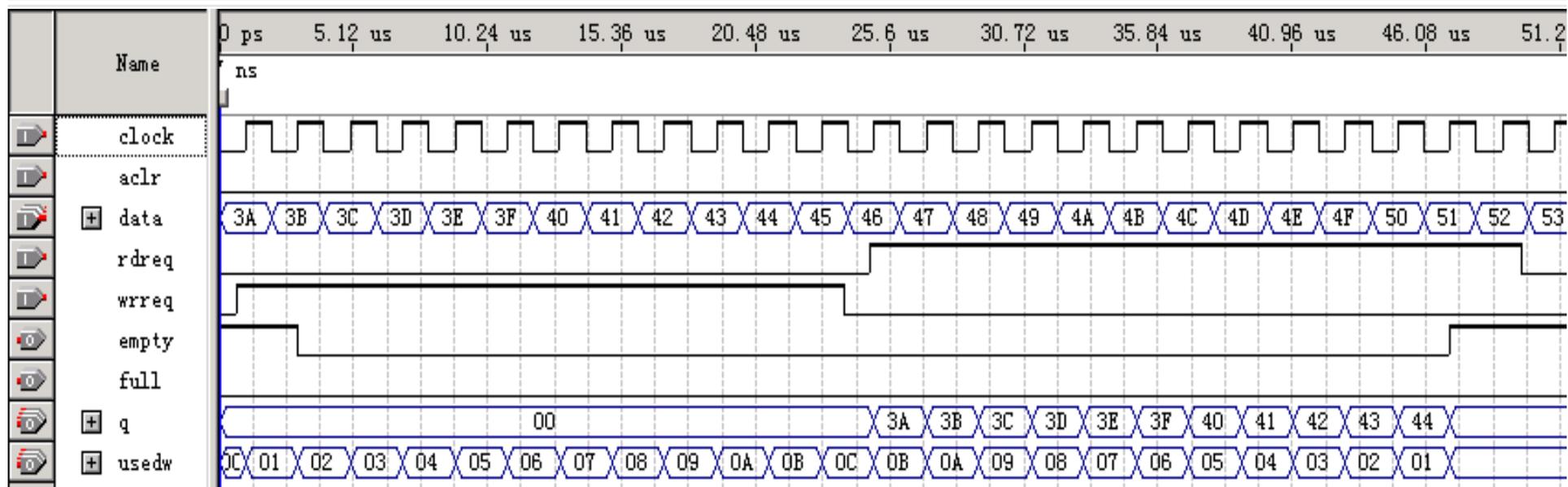


图3-60 FIFO的仿真波形

3.7 嵌入式锁相环ALTPLL调用

3.7.1 建立嵌入式锁相环元件

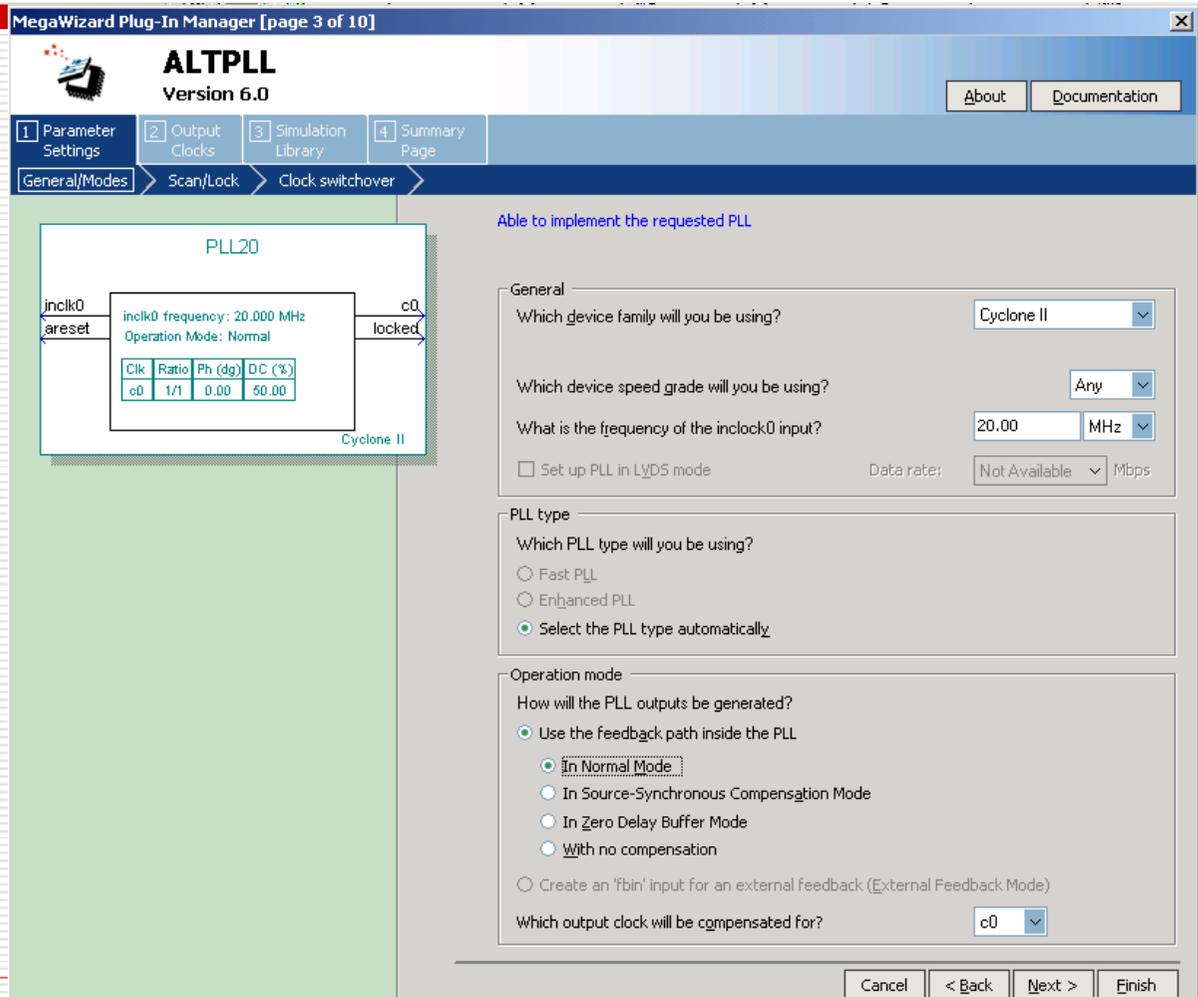


图3-61 选择输入参考时钟为20MHz

3.7 嵌入式锁相环ALTPLL调用

3.7.1 建立嵌入式锁相环元件

PLL20

inclk0
pllena

inclk0 frequency: 20.000 MHz
Operation Mode: Normal

Clk	Ratio	Ph (dg)	DC (%)
c0	7/4	0.00	50.00

c0
locked

Cyclone II

Able to implement the requested PLL

Dynamic configuration

Create optional inputs for dynamic reconfiguration

Optional inputs

Create an 'pllena' input to selectively enable the PLL

Create an 'areset' input to asynchronously reset the PLL

Create an 'pfdena' input to selectively enable the phase/freq. detector

Lock output

Create 'locked' output

Enable self-reset on loss of lock

Hold 'locked' output low for 1048575 cycles after the PLL initializes

Advanced PLL Parameters

Using these parameters is recommended for advanced users only

Create output file(s) using the 'Advanced' PLL parameters

- Configurations with output clock(s) that use cascade counters are not supported

图3-62 选择控制信号

3.7 嵌入式锁相环ALTPLL调用

3.7.1 建立嵌入式锁相环元件

PLL20

inclk0
areset

inclk0 frequency: 20.000 MHz
Operation Mode: Normal

Clk	Ratio	Ph (dg)	DC (%)
c0	3/2	0.00	50.00
c1	5/2	0.00	50.00
c2	10/1	0.00	50.00

c0
c1
c2
locked

Cyclone II

c2 - Core/External Output Clock
Able to implement the requested PLL

Use this clock

Clock Tap Settings

	Requested settings	Actual settings
<input checked="" type="radio"/> Enter output clock frequency:	200.0000000 MHz	200.000000
<input type="radio"/> Enter output clock parameters:		
Clock multiplication factor	1	10
Clock division factor	1	1
Clock phase shift	0.00 deg	0.00
Clock duty cycle (%)	50.00	50.00

图3-63 选择e0的输出频率为200MHz

3.7 嵌入式锁相环ALTPLL调用

3.7.2 测试锁相环

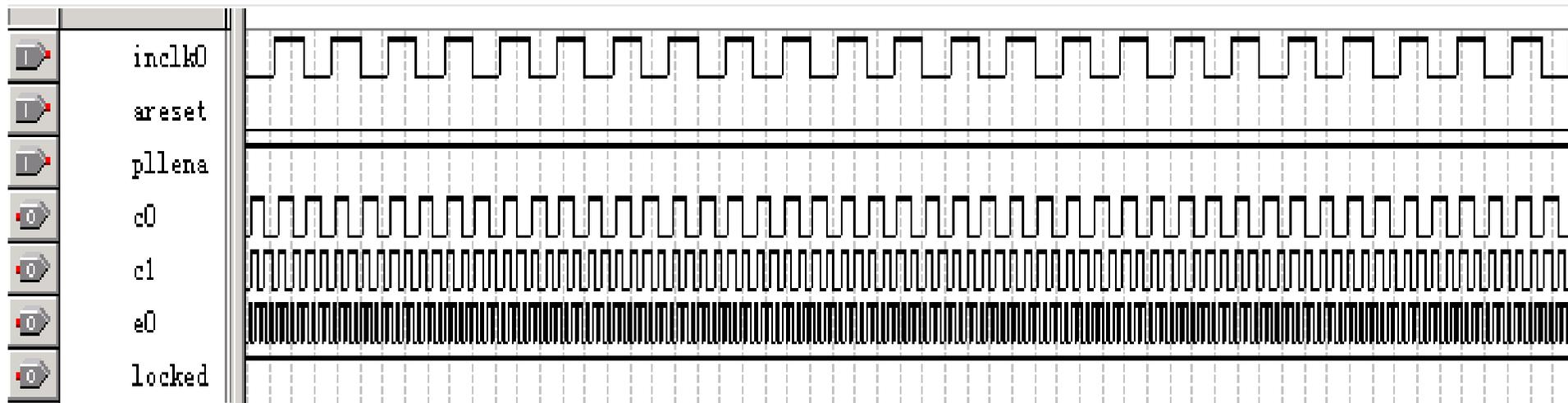


图3-64 ALTPLL元件的仿真波形

3.7 嵌入式锁相环ALTPLL调用

3.7.2 测试锁相环

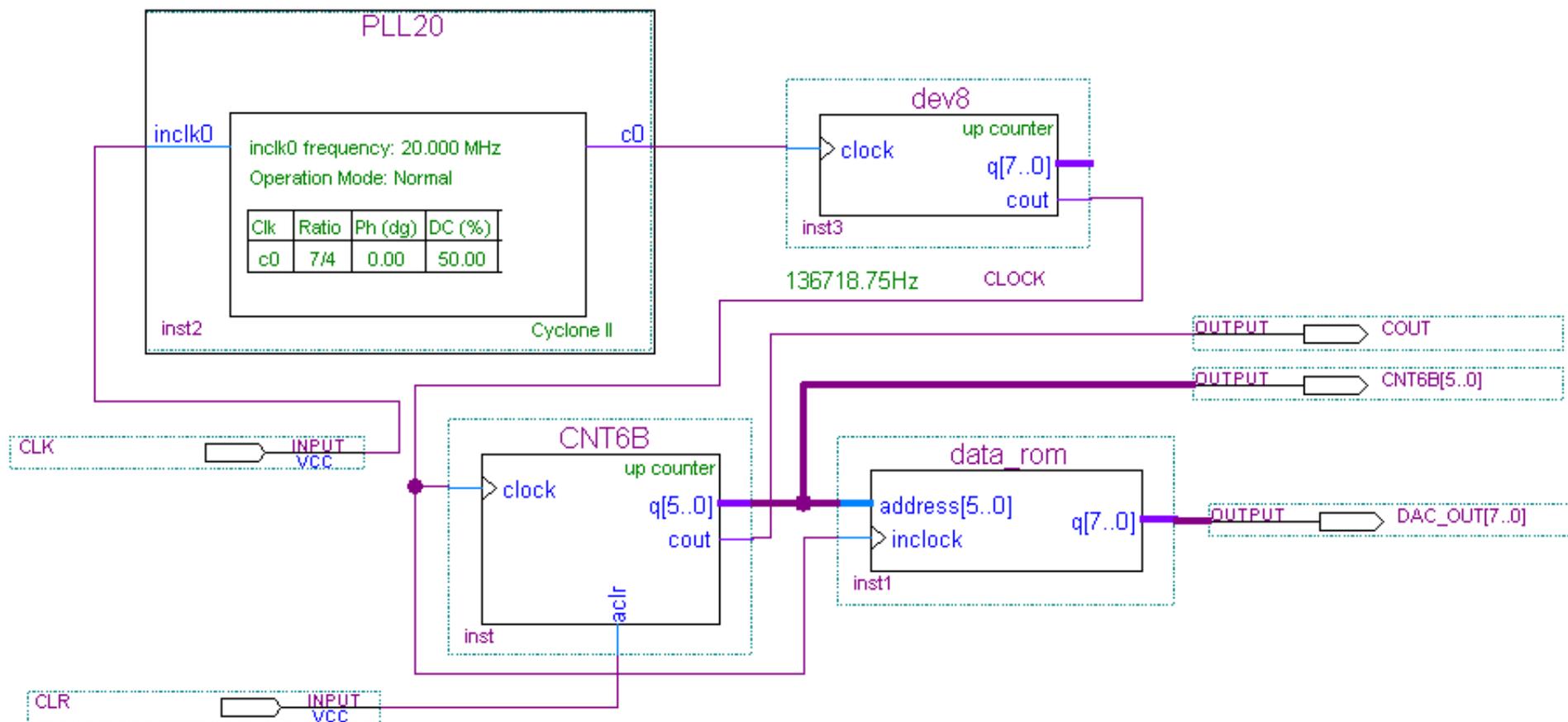


图3-65 增加了锁相环的电路

3.8 优化设计

3.8.1 流水线设计

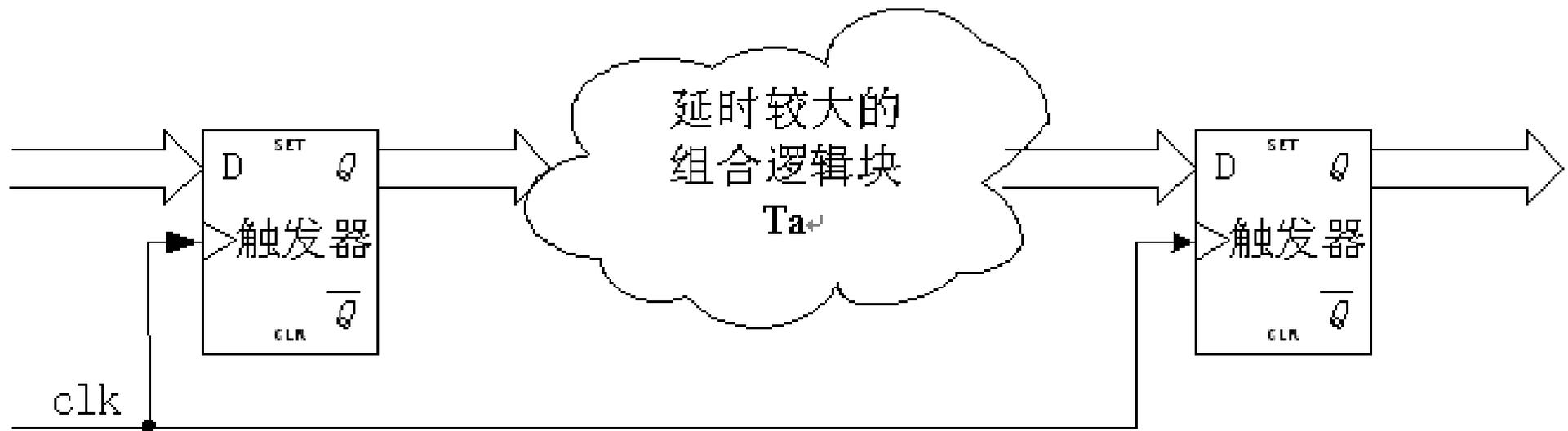


图3-66 未使用流水线

3.8 优化设计

3.8.1 流水线设计

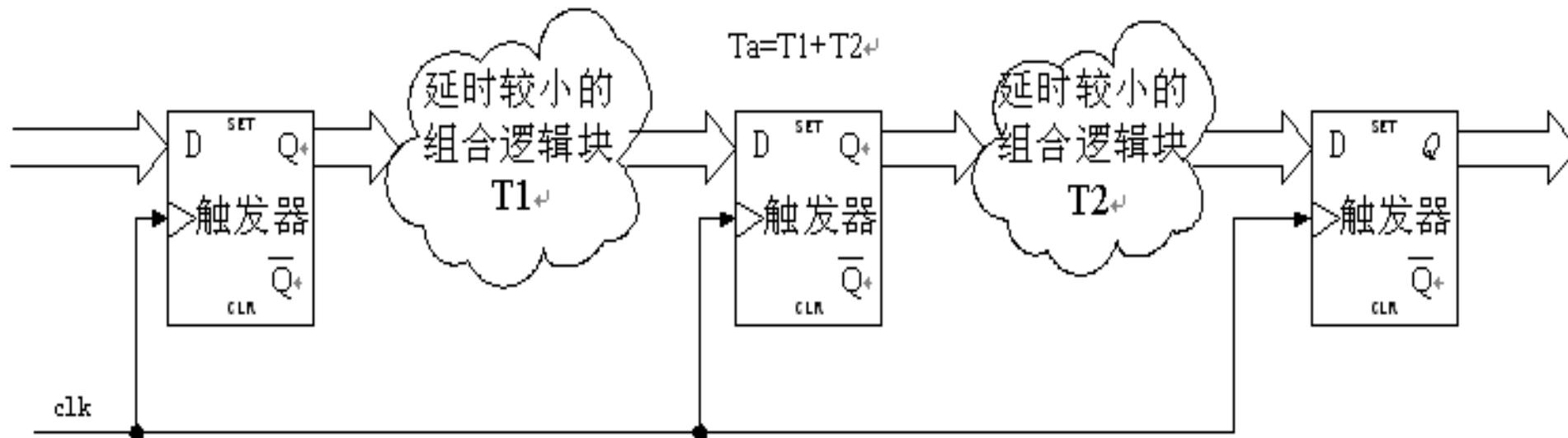
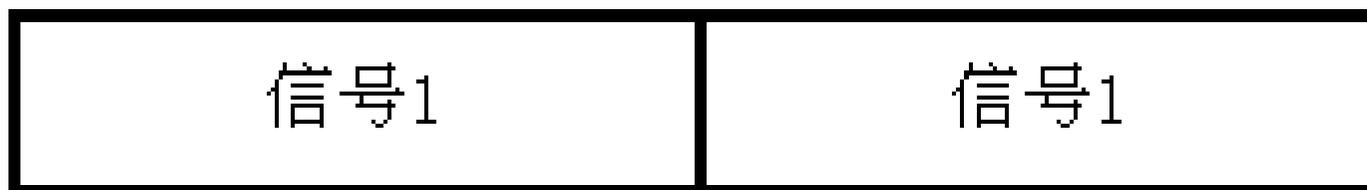


图3-67 使用流水线

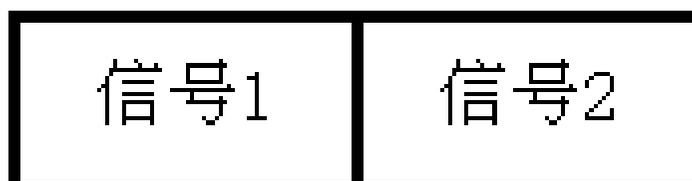
3.8 优化设计

3.8.1 流水线设计

未使用
流水线



流水线
第1级



流水线
第2级

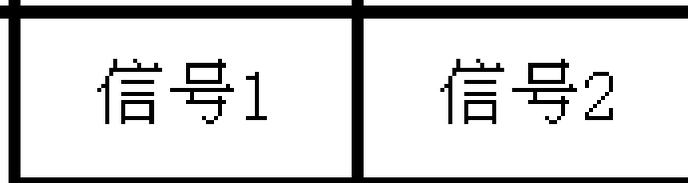


图3-68 流水线工作图示

3.8 优化设计

3.8.2 寄存器平衡技术

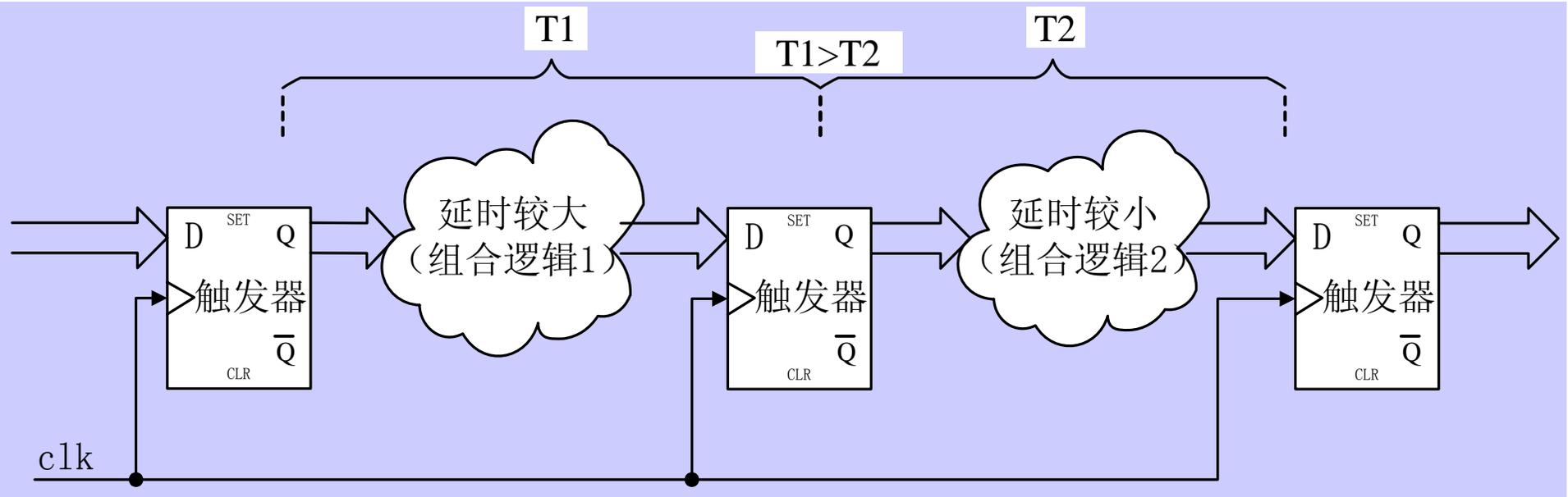


图3-69 不合理的结构

3.8 优化设计

3.8.2 寄存器平衡技术

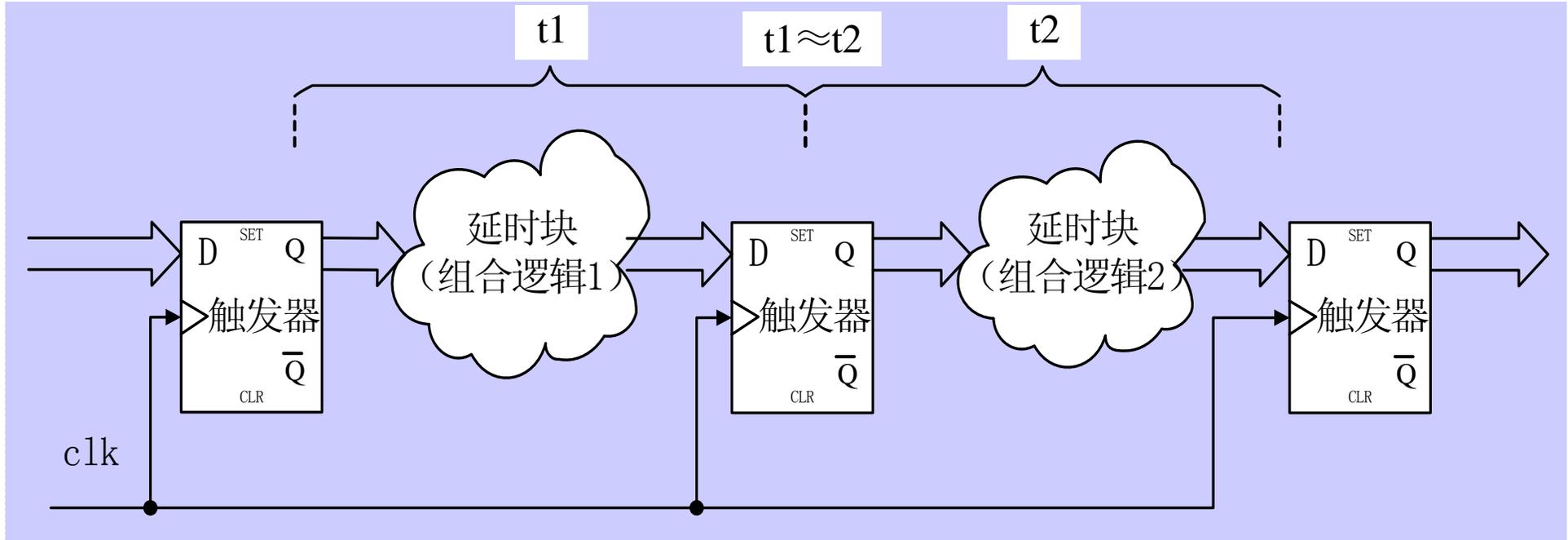


图3-70 寄存器平衡结构

3.9 时序设置与分析

3.9.1 时序约束设置

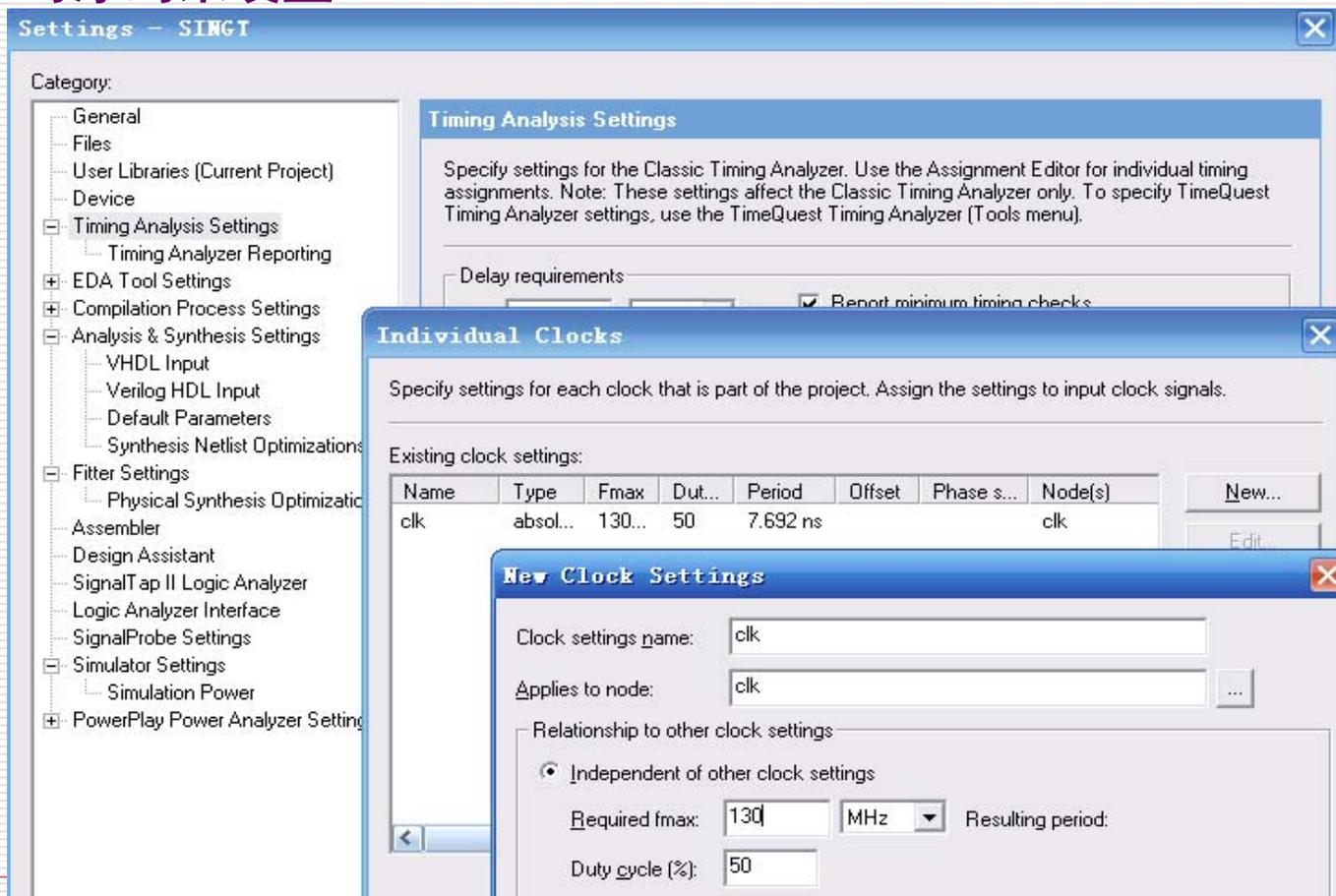


图3-71 全编译前时序条件设置（设置时钟信号CLK不低于130MHz）

3.9 时序设置与分析

3.9.1 时序约束设置

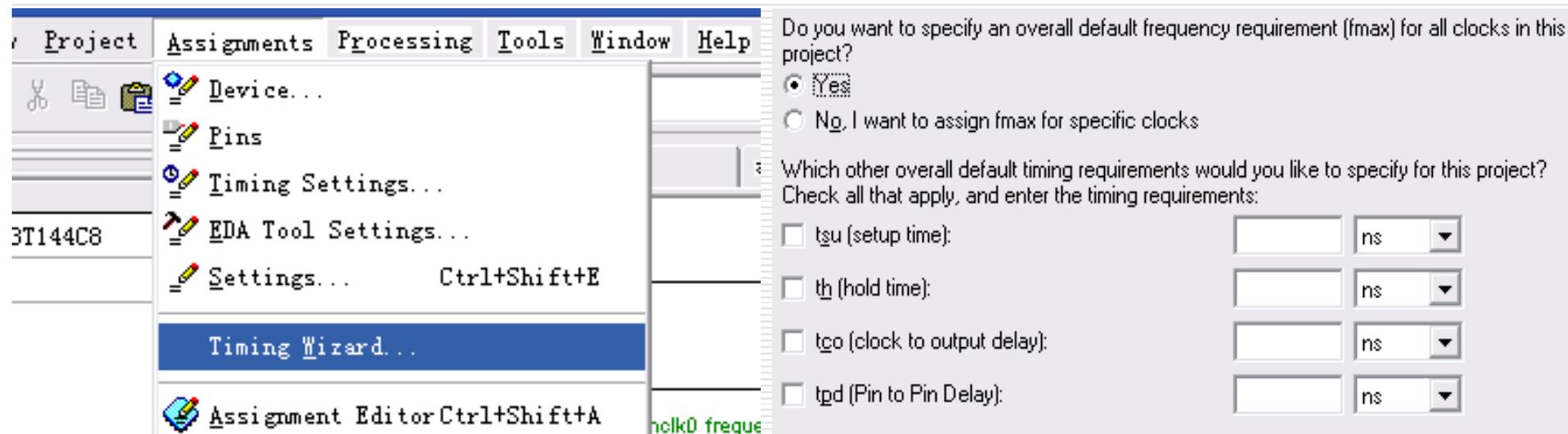
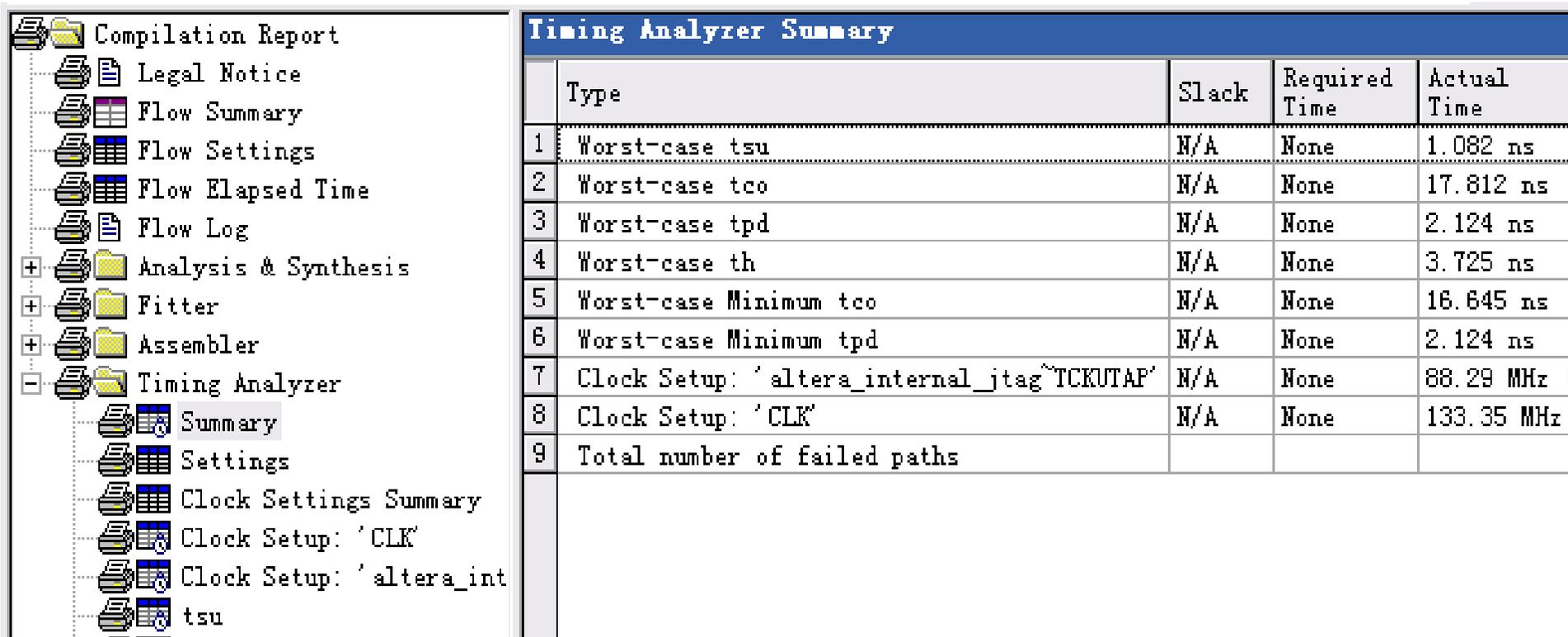


图3-72 由Timing Wizard窗口设置时序条件

3.9 时序设置与分析

3.9.2 查看时序分析结果



Timing Analyzer Summary				
	Type	Slack	Required Time	Actual Time
1	Worst-case tsu	N/A	None	1.082 ns
2	Worst-case tco	N/A	None	17.612 ns
3	Worst-case tpd	N/A	None	2.124 ns
4	Worst-case th	N/A	None	3.725 ns
5	Worst-case Minimum tco	N/A	None	16.645 ns
6	Worst-case Minimum tpd	N/A	None	2.124 ns
7	Clock Setup: 'altera_internal_jtag~TCKUTAP'	N/A	None	88.29 MHz (
8	Clock Setup: 'CLK'	N/A	None	133.35 MHz
9	Total number of failed paths			

图3-73 时序分析报告窗

习题

3-1. 归纳利用Quartus II进行原理图输入设计的流程：从电路编辑输入一直到SignalTap II测试。

3-2. 如何为设计中的SignalTap II加入独立采样时钟？

3-3. 参考Quartus II的Help，详细说明Assignments菜单中Settings对话框的功能。

(1) 说明其中的Timing Requirements & Options的功能、使用方法和检测途径。

(2) 说明其中的Compilation Process的功能和使用方法。

(3) 说明 Analysis & Synthesis Setting 的功能和使用方法，以及其中的 Synthesis Netlist Optimization的功能和使用方法。

(4) 说明Fitter Settings中的Design Assistant和Simulator功能，举例说明它们的使用方法。

3-4. 概述Assignments菜单中Assignment Editor的功能，举例说明。

习题

3-5. LPM_ROM、LPM_RAM、LPM_FIFO等模块与FPGA中嵌入的EAB等模块有怎样的联系？

3-6. 参考QuartusII的Help（Contents），详细说明LPM元件altcam、altsyncram、lpm_fifo、lpm_shiftreg的使用方法，以及其中各参量的含义和设置方法。

3-7. 试归纳QuartusII的In-System Memory Content Editor有那些用途。如果要设计一CPU，如何为它配置含有汇编程序代码的ROM（文件）？

3-8. 试说明，为什么对组合电路加入了流水线结构后，总延时不但没有减少，反而有所增加的情况下，数据的处理速度却能大幅提高？

3-9. 应用流水线结构，为什么要进行寄存器平衡？

实验与实践

3-1. 流水线乘法累加器设计

3-2. 简易逻辑分析仪设计

3-3. 简易正弦信号发生器设计

实验与实践

3-4. 8位16进制频率计设计

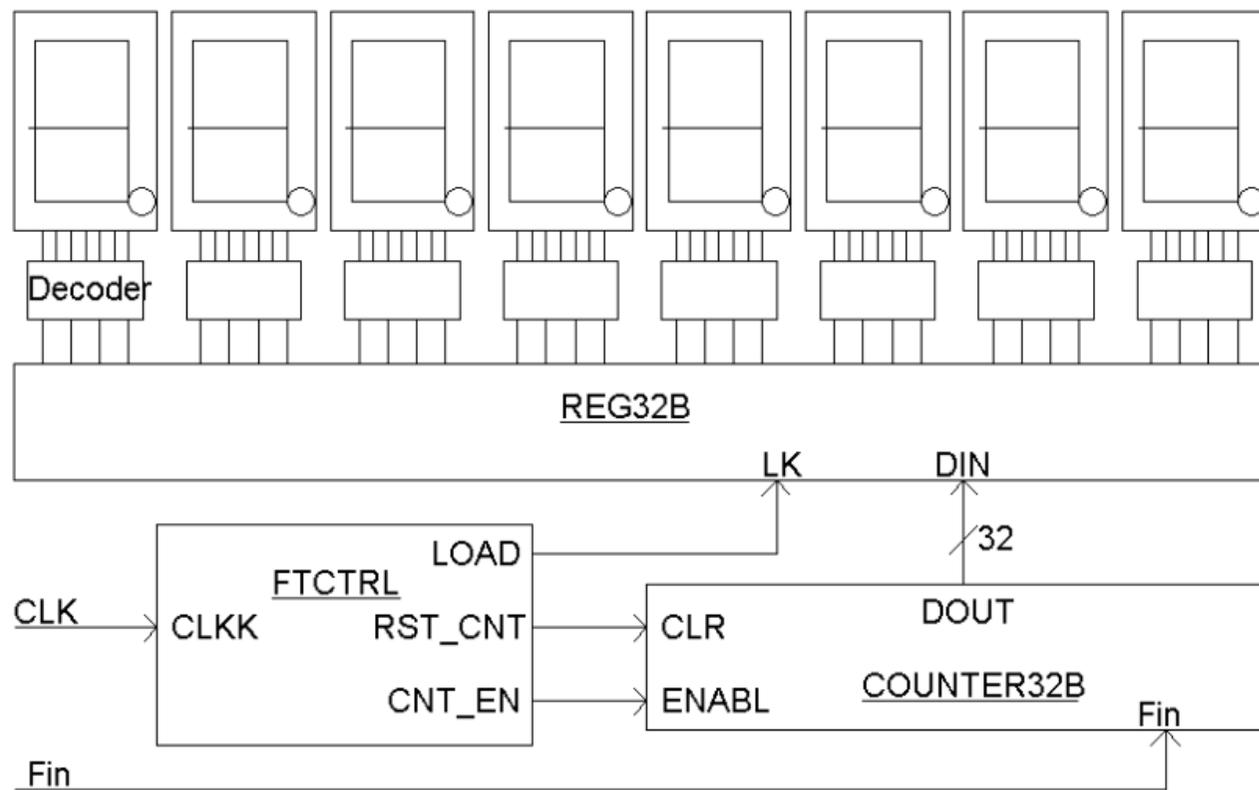


图3-74 频率计电路框图

实验与实践

3-5. 利用LPM_ROM设计乘法器

【例3-3】

```
WIDTH = 8 ;  
DEPTH = 256 ;  
ADDRESS_RADIX = HEX ;  
DATA_RADIX = HEX ;  
CONTENT BEGIN  
00:00 ; 01:00 ; 02:00 ; 03:00 ; 04:00 ; 05:00 ; 06:00 ; 07:00 ; 08:00 ; 09:00 ;  
10:00 ; 11:01 ; 12:02 ; 13:03 ; 14:04 ; 15:05 ; 16:06 ; 17:07 ; 18:08 ; 19:09 ;  
20:00 ; 21:02 ; 22:04 ; 23:06 ; 24:08 ; 25:10 ; 26:12 ; 27:14 ; 28:16 ; 29:18 ;  
30:00 ; 31:03 ; 32:06 ; 33:09 ; 34:12 ; 35:15 ; 36:18 ; 37:21 ; 38:24 ; 39:27 ;  
40:00 ; 41:04 ; 42:08 ; 43:12 ; 44:16 ; 45:20 ; 46:24 ; 47:28 ; 48:32 ; 49:36 ;  
50:00 ; 51:05 ; 52:10 ; 53:15 ; 54:20 ; 55:25 ; 56:30 ; 57:35 ; 58:40 ; 59:45 ;  
60:00 ; 61:06 ; 62:12 ; 63:18 ; 64:24 ; 65:30 ; 66:36 ; 67:42 ; 68:48 ; 69:54 ;  
70:00 ; 71:07 ; 72:14 ; 73:21 ; 74:28 ; 75:35 ; 76:42 ; 77:49 ; 78:56 ; 79:63 ;  
80:00 ; 81:08 ; 82:16 ; 83:24 ; 84:32 ; 85:40 ; 86:48 ; 87:56 ; 88:64 ; 89:72 ;  
90:00 ; 91:09 ; 92:18 ; 93:27 ; 94:36 ; 95:45 ; 96:54 ; 97:63 ; 98:72 ; 99:81 ;  
END ;
```

实验与实践

3-6 简易存储示波器设计

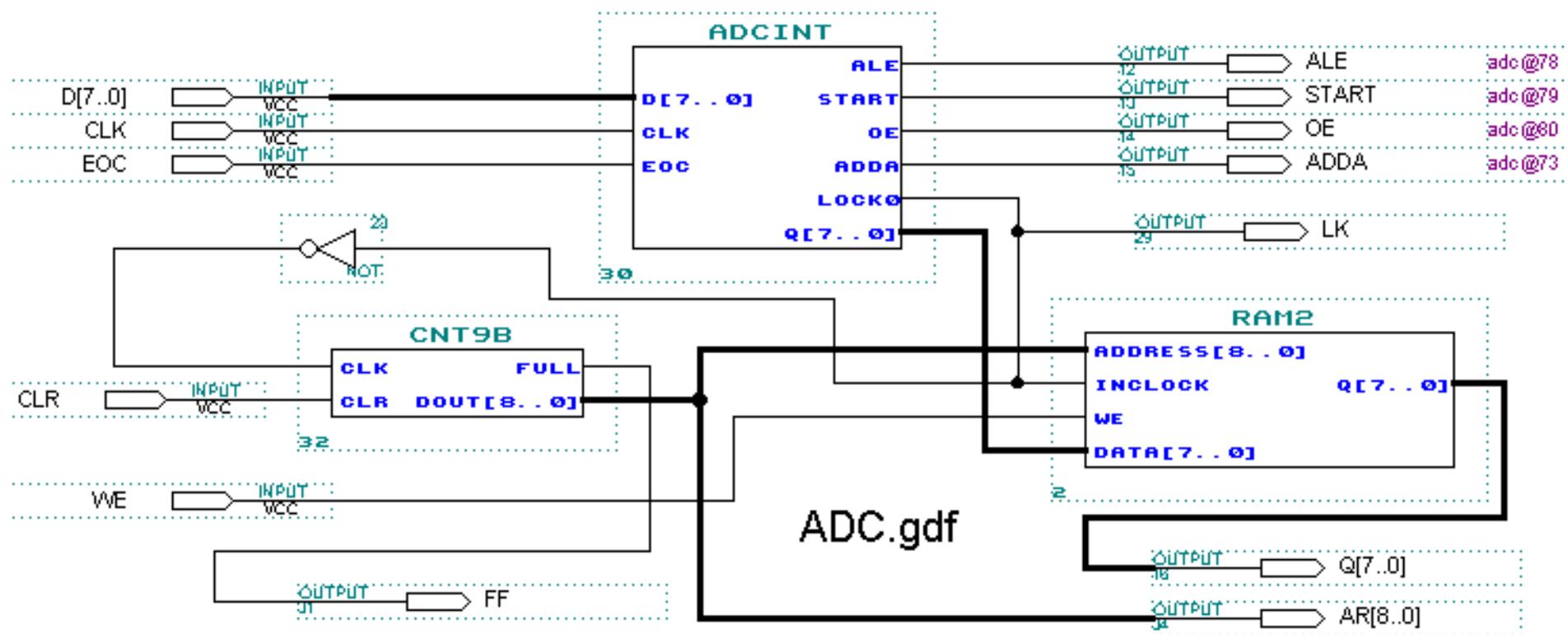


图3-75 简易存储示波器电路图

实验与实践

3-7 LPM_FIFO实验

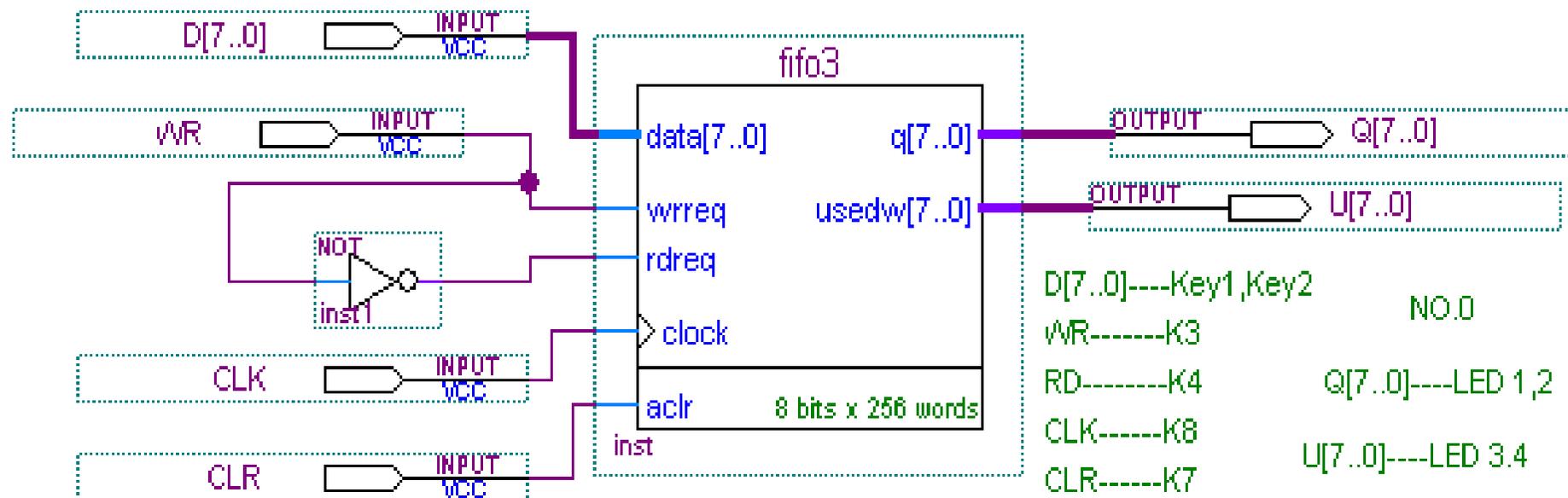


图3-76 lpm_fifo的实验结构图

实验与实践

3-7 LPM_FIFO实验

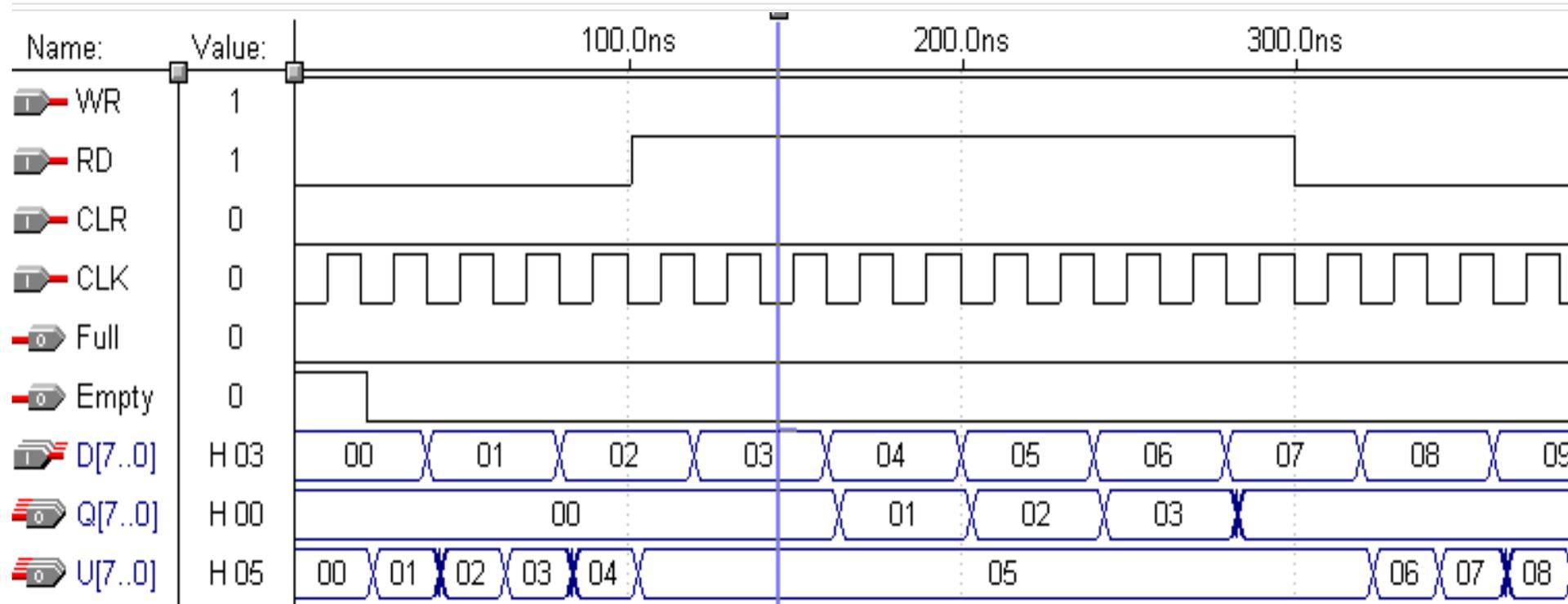


图3-77 lpm_fifo的仿真波形图